LARY_CR: Software package for the Analysis of Cardio Vascularand Respiratory Rhythms, in the SCILAB_SCICOS environment

Claire Médigue, Alessandro Monti, Aurélie Wambergue

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

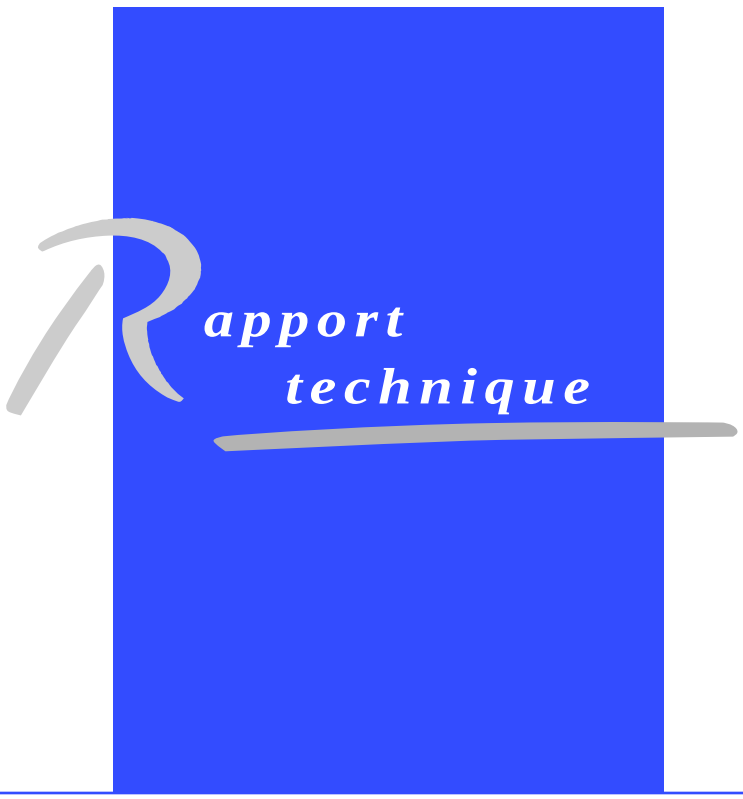
LARY_CR: Software package for the Analysis of Cardio Vascular and Respiratory Rhythms, in the SCILAB_SCICOS environment

Claire Médigue — Alessandro Monti — Aurélie Wambergue

N° 0259

April 10, 2002

THÈME 4



ISSN 0249-0803

LARY_CR: Software package for the Analysis of Cardio Vascular and Respiratory Rhythms, in the SCILAB_SCICOS environment

Claire Médigue , Alessandro Monti , Aurélie Wambergue

Thème 4 — Simulation et optimisation
de systèmes complexes
Projet SOSSO

Rapport technique n° 0259 — April 10, 2002 — 88 pages

Abstract: LARY_CR is a software package dedicated to the study of cardiovascular and respiratory rhythms, developed in the SCILAB_SCICOS scientific environment. It presents signal processing methods, from events detection on raw signals to the variability analysis of the resulting time series. The events detection concerns the heart beat recognition on the electrocardiogram, defining the RR time series, the maxima and minima on the arterial blood pressure defining the systolic and diastolic time series. These detections are followed by the resampling of the time series then their analyse. This analyse uses temporal and time frequency methods: Fourier Transform, spectral gain between the cardiac and blood pressure series, Smooth Pseudo Wigner_Ville Distribution, Complex DeModulation, temporal method of the cardiovascular Sequences.
This software is aiming to give some tools for studying the autonomic nervous system, acting in particular in the baroreflex loop; its functionning is reflected by the cardiovascular variabilities and their relationships with the other physiological signals, especially the respiratory activity.

Key-words: cardiovascular system, autonomic nervous system, signal processing

Unité de recherche INRIA Rocquencourt
Domaine de Voluceau, Rocquencourt, BP 105, 78153 Le Chesnay Cedex (France)
Téléphone : +33 1 39 63 55 11 —Télécopie : +33 1 39 63 53 30

LARY_CR: Logiciel d'Analyse des Rythmes Cardiovasculaires et Respiratoires

Résumé : LARY_CR est un logiciel d' analyse des rythmes cardio-vasculaires et respiratoires, développé dans l'environnement d'analyse numérique SCILAB_SCICOS. Il propose diverses méthodes de traitement du signal, allant de la détection d´évènements sur les signaux bruts jusqu'à l'analyse de la variabilité des séries temporelles résultantes. La détection d'évènements concerne la reconnaissance des battements cardiaques sur l'électro-cardiogramme, définissant la série des intervalles RR, des maxima et minima du signal de pression artérielle, définissant les séries systoliques et diastoliques. Elle est suivie du réechantillonnage des séries temporelles obtenues puis de leur analyse. Cette analyse utilise des méthodes temps et temps-fréquence: Transformée de Fourier, Gain spectral entre les séries cardiaques et de pression artérielle, Distribution de Pseudo Wigner-Ville lissée, Démodulation Complexe Modifiée, méthode des séquences cardio-vasculaires. L'objectif de ce logiciel est de donner des moyens d'étude du système nerveux autonome, qui intervient en particulier dans la boucle baroréflexe, et dont l'état est reflété par les variabilités des rythmes cardio-vasculaires et leurs relations avec d'autres signaux physiologiques, notamment la respiration.

Mots-clés : système cardio-vasculaire, système nerveux autonome, traitement du signal

Contents

1 Introduction **5**
1.1 Physiological Introduction 5
1.1.1 The autonomic nervous system and the baroreceptor control loop 5
1.1.2 Cardiovascular and respiratory time series 5
1.1.3 RR and blood pressure variability 6
1.2 The SCILAB/SCICOS scientific environment 7
1.2.1 The choices 7
1.2.2 SCICOS program: an example 8
1.3 LARY_CR: general presentation 13

2 Signal Preprocessing: Finite Impulse Response filters **14**
2.1 Choice of the Finite Impulse Response filters 14
2.2 Examples of Finite Impulse Response filters, adapted to CV series 15

3 Obtention of cardio-vascular and respiratory time series **20**
3.1 Presentation 20
3.2 Detection processing 20
3.2.1 Principle of a detection program 20
3.2.2 RR time series 22
3.2.3 Pulse Interval, Systolic and Diastolic time series 27
3.2.4 Respiratory signal 32
3.3 Validation of the detection processing and resampling of the time series 33
3.3.1 RR time series 33
3.3.2 Pulse Interval, Systolic and diastolic time series 36
3.3.3 Respiratory signal 38
3.4 Description of the systolic and diastolic research algorithm 39
3.4.1 The *sysdiapeak.f* function 39
3.4.2 La fonction *sysdiapeak.f* 40

4 Spectral analysis of cardiovascular and respiratory time series **43**
4.1 Presentation 43
4.2 Fast Fourier Transform 43
4.2.1 Introduction to the frequential smoothing 43
4.2.2 Smoothed Power Spectral Density assessment in the whole spectrum . . . 43
4.2.3 SPSD and spectral Dispersion assessment in a frequency band 43
4.3 Spectral Baroreflex Sensitivity: Coherence and Gain assessment 53
4.3.1 Spectral Coherence and Gain assessment on the whole spectrum 53
4.3.2 Spectral Coherence and Gain assessment in a frequency band 56
4.3.3 The whole spectral Gain application 56

5 Time-varying analysis of cardiovascular and respiratory time series **59**
5.1 Presentation 59
5.2 Complex Transformation of the cardiovascular time series 59
5.3 Complex DeModulation (CDM) 63
5.3.1 Description of the blocks 63

5.3.2 The whole CDM application 64
5.4 Smoothed Pseudo Wigner-Ville distribution (SPWVD) 70
5.4.1 The SCICOS implementation of the SPWVD 70
5.4.2 The 3D visualisation of the SPWVD 77

6 Time analysis of cardiovascular time series **82**
6.1 Presentation 82
6.2 The Sequences Method 82

7 Practical aspects of LARY_CR **85**
7.1 The directories hierarchy 85
7.2 Install 86
7.3 How to create or modify a program in the SCICOS environment? 88
7.3.1 At the SCICOS graphical level 88
7.3.2 At the F and C functions level 88

1 Introduction

This Technical Report describes LARY_CR (*logiciel d'analyse des rythmes - cardiovasculaires et respiratoire*), a signal processing toolbox, developed in the scientific environment (SCILAB/SCICOS) and dedicated to the analysis of cardiovascular and respiratory rhythms.

It provides methods to detect the events of interest from the raw cardiovascular (CV) signals, such as the R peak from the electro-cardiogram; and methods to analyze the short term cardiovascular variabilities, related to the autonomic nervous system (ANS). Classical signal processing methods, such as spectral decomposition or time-frequency representations, are adapted to estimate respiratory and CV interactions.

Several studies were already assessed using LARY_CR, under physiological (orthostatisme, sleep), pathological (chronic heart failure, vasovagal syncope) and pharmacological conditions (activator/inhibitor of the ANS).

This Technical Report refers to the INRIA Research Report, "Short-term control of the cardiovascular system: modelling and signal analysis" n^o4427 (http://www.inria.fr/rrrt/rr-4427.html), which presents the theorical aspects of the implemented methods and their applicaticon to clinical studies. Therefore, only a short physiological introduction will follow to explain our domain of interest.

This Technical Report refers to the SCILAB/SCICOS free scientific environment, developed at INRIA and accessible at http://www-rocq.inria.fr/scilab/. All the theorical concepts about this environment, the predefined functions and examples will be found at this adress.

1.1 Physiological Introduction

1.1.1 The autonomic nervous system and the baroreceptor control loop

The cardiovascular system (CV) is under control mechanisms which have different dynamics and we are interested only in the short term control, about few minutes, assumed by the autonomic part of the nervous system (ANS). This control involves fast mechanisms of blood flow and pressure regulation, the baroreceptor control loop, neglecting slower ones, such as hormonal regulation. It acts through sympathetic and parasympathetic neural innervation, which have an antagonistic effect on the heart rate: the sympathetic is an accelerator and the parasympathetic an inhibitor. Therefore, the heart rate is the result of the autonomic balance on the cardiac pacemaker firing rate. In addition, the sympathetic component has a contractile force effect of the ventricle and a vasoconstrictor effect on the vessels.

1.1.2 Cardiovascular time series

Only cardiac beat to beat variables extracted from raw signals are used and not the raw signals themself, as explained in the Research Report. A typical pattern of blood pressure, respiratory and electrocardiographic signals is shown in fig. (1). They constitute the following time series: RR interval, systolic blood pressure (SBP), diastolic blood pressure (DBP).
The ECG represents the electrical events of depolarisation and repolarisation of the cardiac

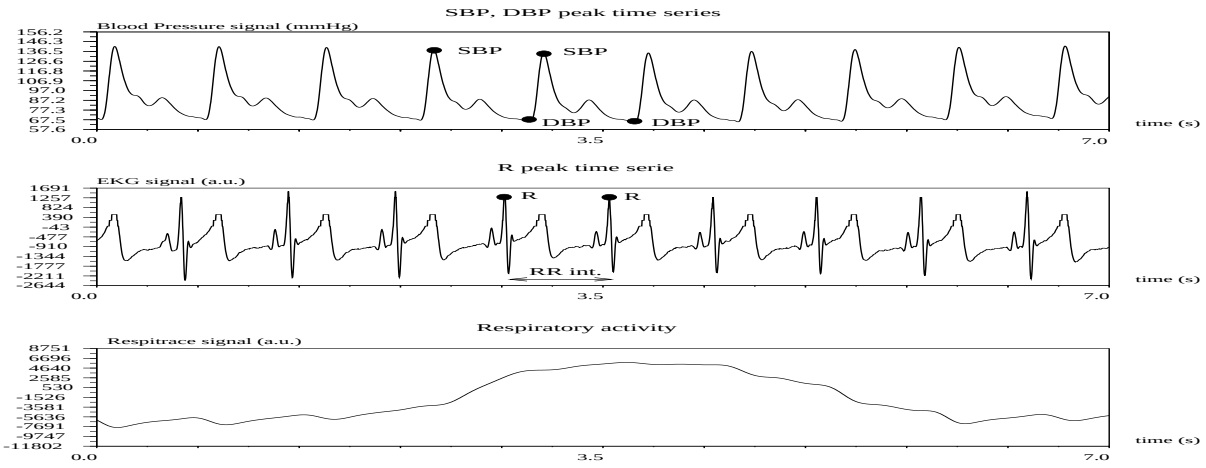


Figure 1: *Blood pressure, electrocardiographic and respiratory raw signals obtained respectively by Finapres, thoracic electrodes and respitrace.*

atrium and ventricles, preceding the mechanical events of contraction and ejection of blood in arterial vessels. To analyse the effects of the ANS on the sinus node, located in the atrium, the best way should be to detect the P wave, corresponding to the atrium depolarisation. But the R peak, corresponding to the maximum of the ventricular depolarisation, is the easiest event to detect and usually accepted to represent the global effect of the ANS on the cardiac frequency.. So, the RR intervals series, expressed in ms, represents the time interval length between two consecutive heart beats; its inverse, the heart rate (HR) is expressed in beats/mn. We analyse only the RR time series, as discussed in the Research Report.
SBP and DBP rhythms are the successive amplitudes of blood pressure corresponding to mechanical cardiac events: SBP is the maximum value of the pressure, corresponding to the ejection of the blood into the aorta, DBP is the minimal value of the pressure, occurring during the cardiac diastole. The Pulse Interval (PI) represents the time interval length between two consecutive systolic peaks.
The respiratory rhythm is slower than the cardiac rhythm and a respiratory cycle can contain approximately from three to ten heart beats. The respiratory activity is the more often measured by oronasal thermistor, thoracic and abdominal movement sensors, which cannot give a good estimation of the respiratory volume. From the respiratory signal, we analyse only the breathing rate by frequential or time-frequential analysis.

1.1.3 Autonomic nervous system, RR interval and blood pressure variability

The mean cardiovascular values, over few minutes, vary according to different physical conditions, such as rest and exercise, but also the cardiovascular values fluctuate around the mean value under relative stable conditions. This dynamic behavior, called cardiovascular variability, implies that cardiovascular mean values and variability have to be studied together to reflect the ANS short-term control of the cardiovascular system. Cardiovascular fluctuations can be studied through beat-to-beat arterial blood pressure and RR means and variances and, as has more recently been

done, through frequency domain analysis. RR power spectrum is considered to be an effective non invasive measure for studying neural modulations of RR in many clinical and research studies.

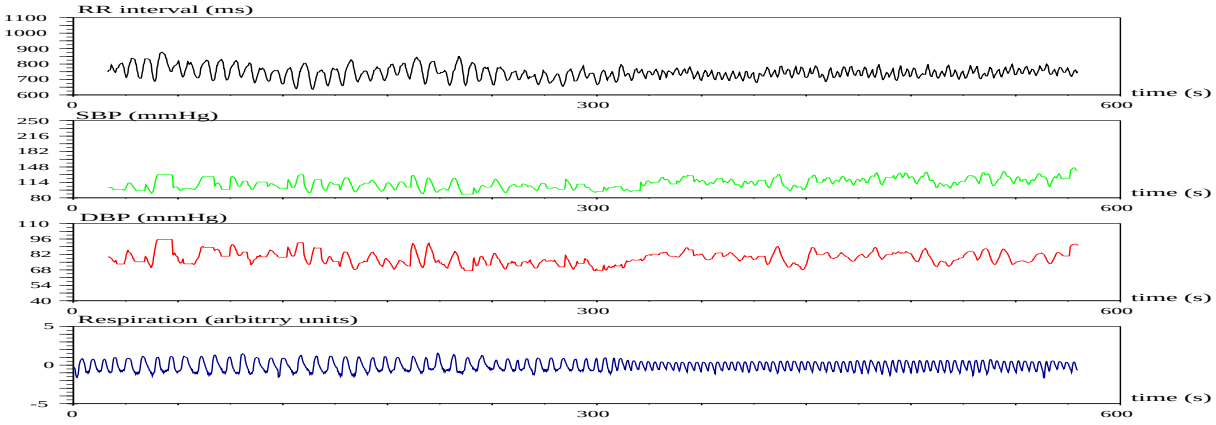


Figure 2: *RR interval, SBP, DBP and respiratory time series with a 2 Hz resampling.*

But more precise characterizations of RR control have been performed by relating RR variability (RRV) to respiration and arterial blood pressure variability (BPV). Three major oscillatory components are usually detectable in both RRV and BPV with more or less pronounced spectral peaks:

1. the highest, synchronous with respiration (the so called respiratory sinus arrhythmia, RSA, in RR signal, and respiratory oscillations in blood pressure signal), is described as "*high frequency*" (HF, about 0.25Hz and varying with respiratory frequency);

2. the spectral peak in the middle, which corresponds to slow waves (or Mayer waves) is described as "*low frequency*" (LF, about 0.1Hz). However, the central frequency of the LF component can also vary considerably (from 0.04 to 0.13Hz)

3. the slowest, corresponding to the very slow waves is described as "*very low frequency*" (VLF, from 0 to 0.04Hz).

We are interested only in LF and HF spectral peaks, which reflect short term control of the cardiovascular system.

1.2 The SCILAB/SCICOS scientific environment

1.2.1 The choices

Scilab is a scientific software package for numerical computations in a user-friendly environment. Among it useful features, we were interested particularly in:

- Hundreds of built-in math functions (new primitives can easily be added).
- Open structure (easy interfacing with Fortran and C via online dynamic link).
- Signal processing libraries

- Sophisticated interpreter and programming language with Matlab-like syntax.
- Scicos, an interactive environment for modeling and simulation of dynamical systems.

We use the SCICOS as well as the SCILAB programing: the SCICOS graphical programs were choosen in a pedagogical objectif for medical users, because it evidences the flow chart processing and makes parameters easely adjustable. A SCICOS application is easy to adapt to different kinds of clinical applications; morever, it is interfaced with C or F libraries, that allows to create new functions and tu use the existing ones
The SCILAB programs were choosen for applications without pedagogical interest or when we wanted to use some predefined functions such as the splines for the resampling.

1.2.2 SCICOS program: an example

A simple example of a SCICOS diagram is illustrated on the figure 3. It computes the means and standard_deviations of an RR signal, plots the raw RR and the means on graphic windows and writes means and standard_deviations on an output file. It shows different elementary SCICOS blocks and their dialog panels, where are defined the parameters. Numeric values can be affected to them through the general context. In SCICOS, all time related values are expressed in seconds. These dialog panels are the interface with the library of C or fortran functions. The function of interface (or drawing function) as well as the simulation function can be identified by clicking in the **object** menu of the SCICOS'main window on **Get Info**, as illustrated on the figure 4. The predefined blocks, as well as the cardiovascular and respiratory blocks (CVR_blocks), are stocked in palettes and can be used to built new applications. These palettes can be selected by clicking **Palettes** in the **Edit** menu (see Fig 5).

CVR_blocks: *mean StDev* computes the means and standard deviations of the input signal on fixed windows of "npoints". *Time signal* computes the time in seconds for each resampled RR.

Input-output blocks: *read from input file* and *write to output file* are used for ascii files under different formats. The figure 6 details the *write to output file* predefined block, because it presents a particularity: the first column of the output file is dedicated to the SCICOS simulation clock. This explains that the input size is 2, for the RR means and standard-deviations and that the output format is 3, for the SCICOS simulation clock, the RR means and standard-deviations.
The two *clocks* send events for the activation of the blocks. The reading clock acts with the sampling frequency of the input data; the calculation clock acts each time a mean has to be computed. The two *scopes* plot the raw RR signal at the reading rhythm and the mean RR signal at the calculation rhythm.

branching blocks: The *Mux* is a multiplexer, allowing to adapt the two outputs of the *mean StDev* block to the one input of the *write to output file* block.

Events block The *STOP* block has a unique activation, at the second 600, as notified in the *Event at time 600.*

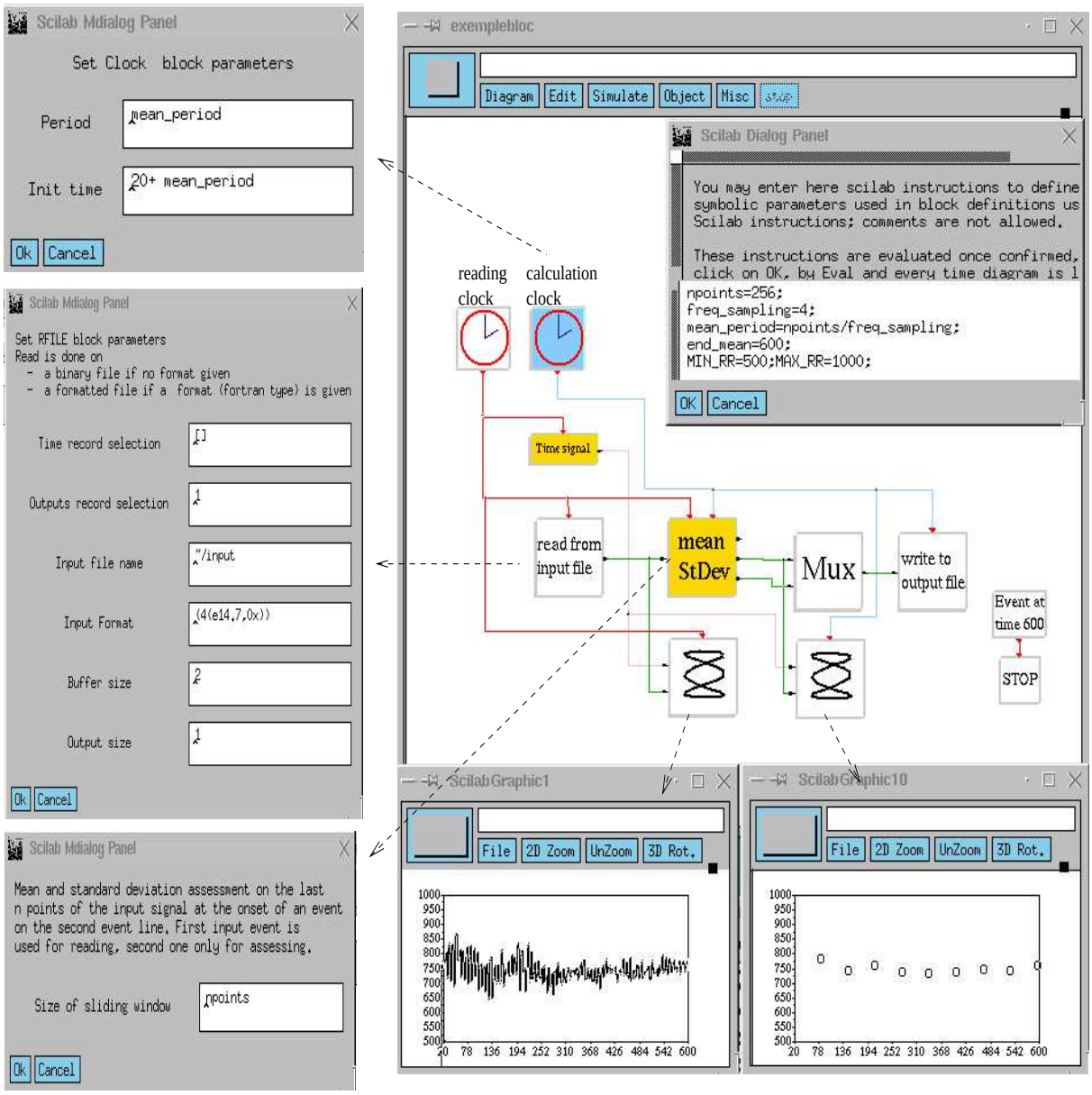


Figure 3: *Example of a SCICOS program in its graphic environment, with some inputs-outputs, calculation, branching, events blocks and their context.*

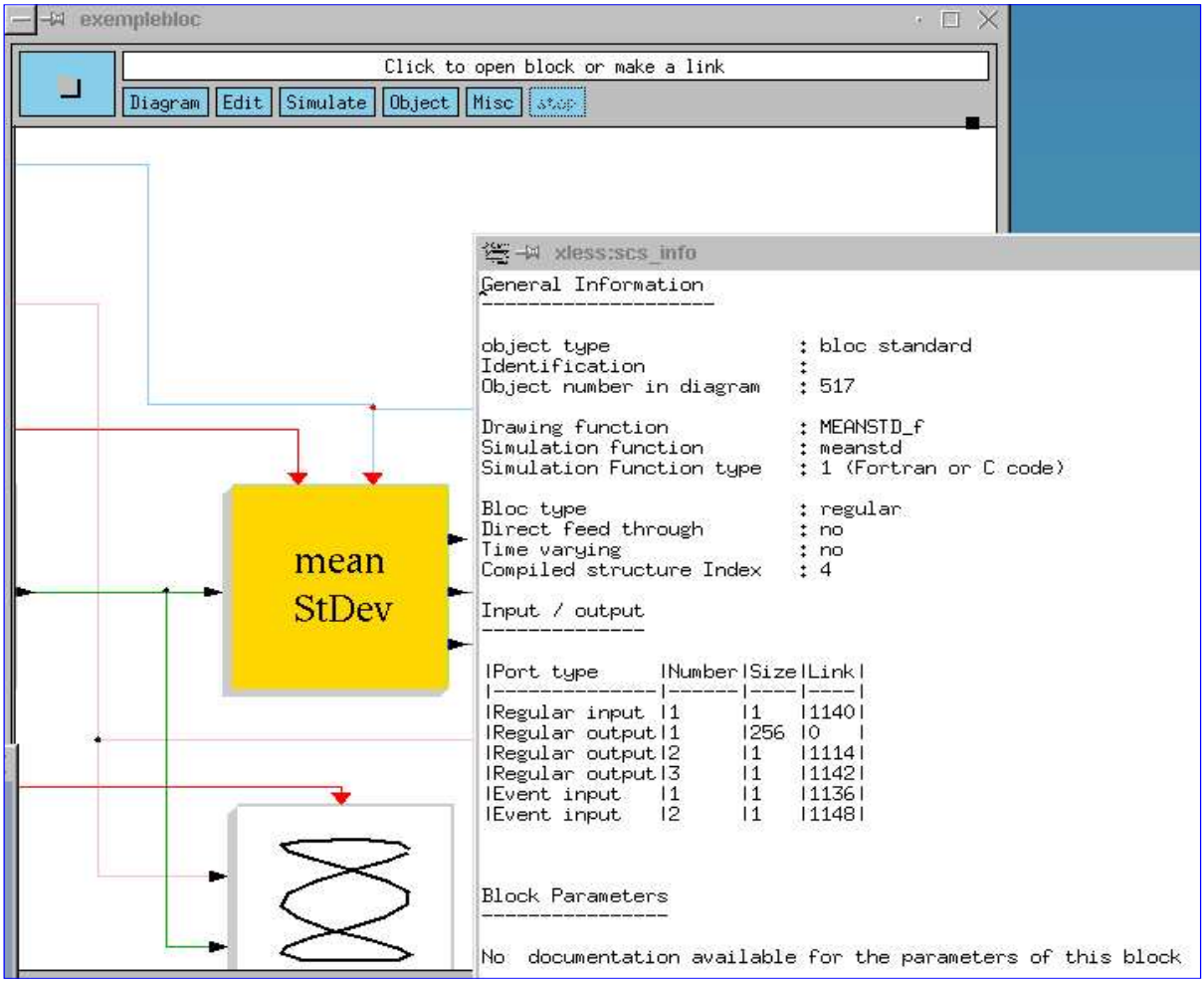


Figure 4: *General information on a SCICOS block*

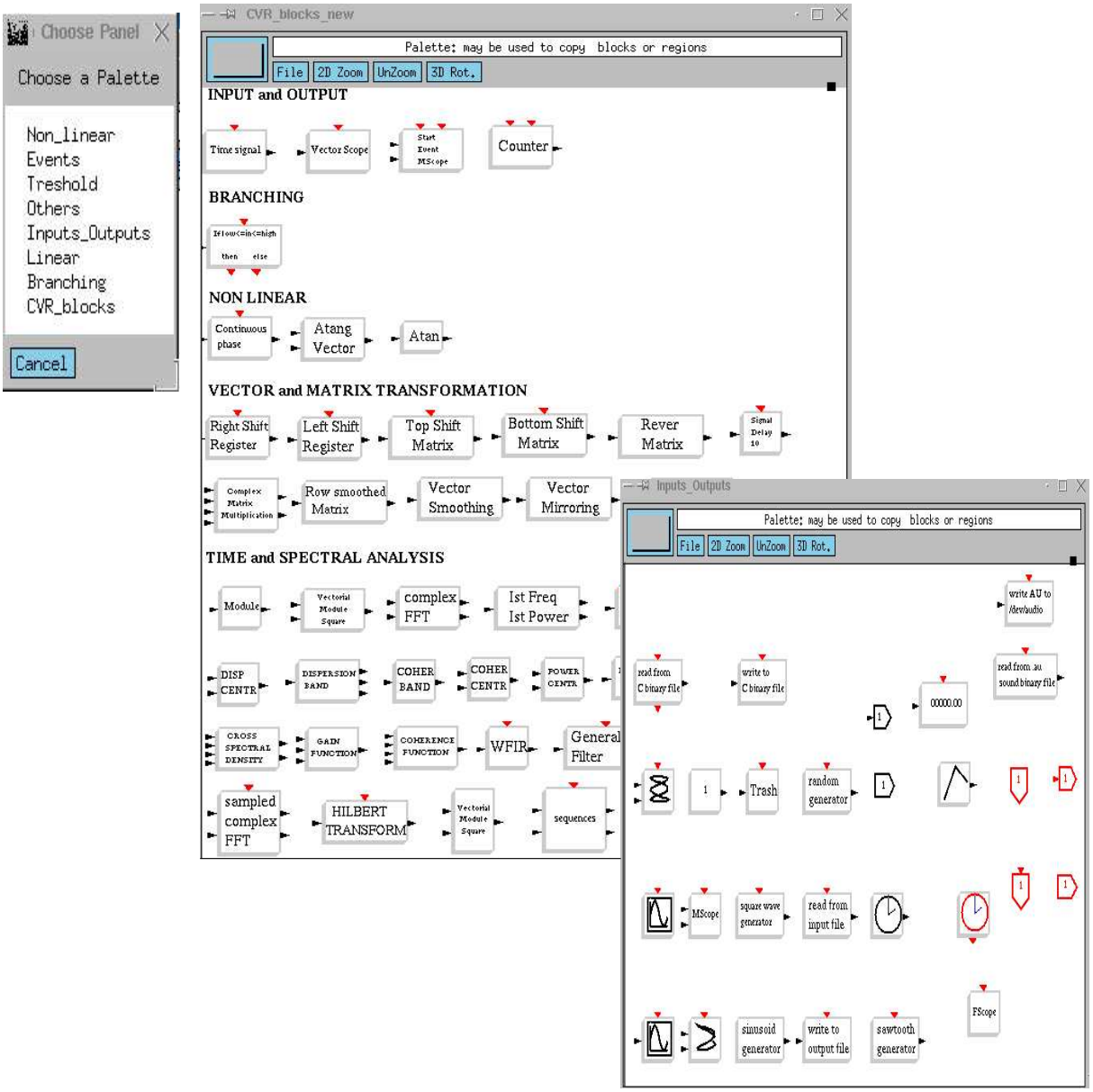


Figure 5: Example of two palettes: the CVR palette, dedicated to LARY_CR applications, and the Inputs_Outputs palette, predefined in SCICOS

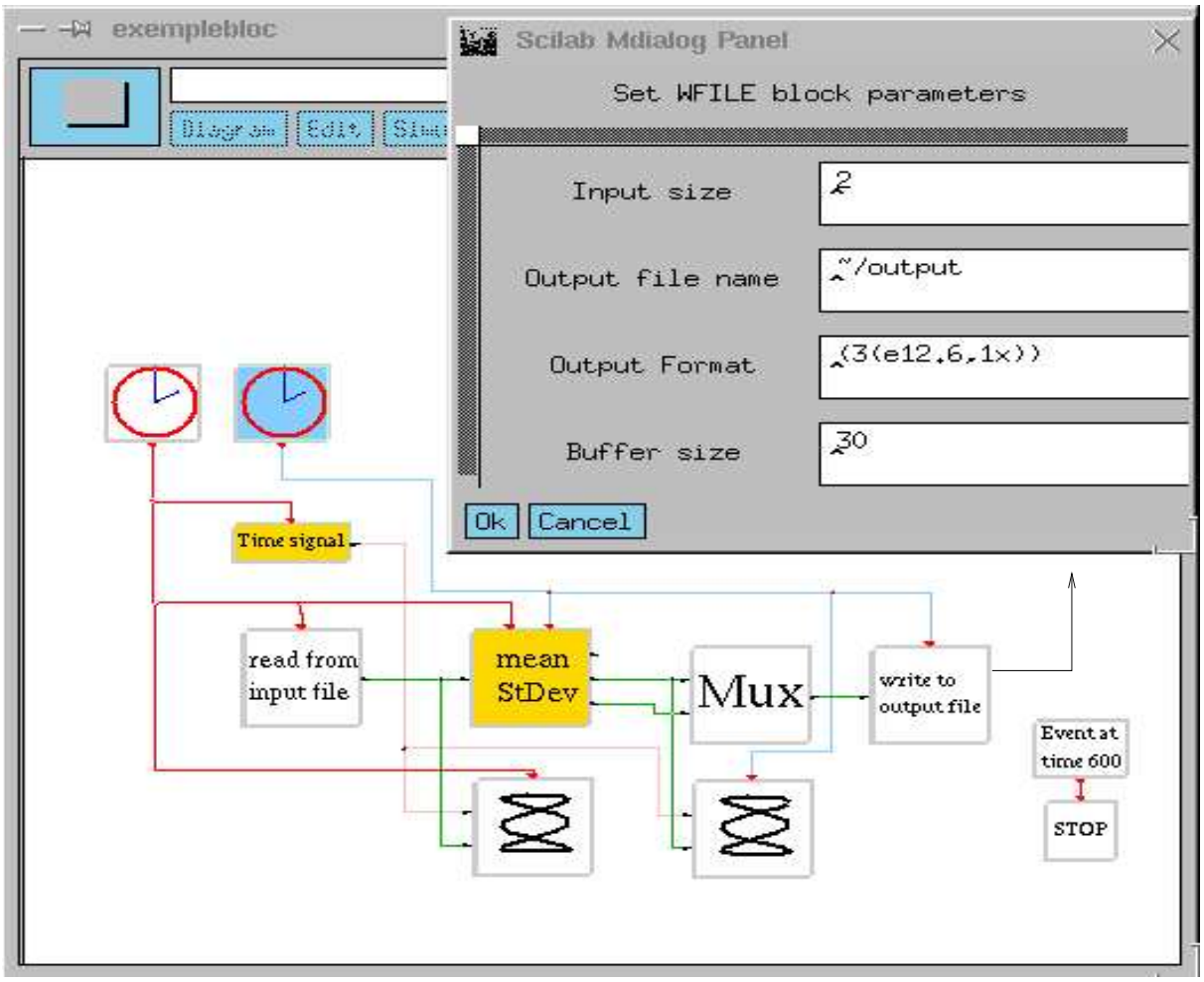


Figure 6: *Detail about the* write to output file *SCICOS predefined function*

1.3 LARY_CR: general presentation

Two major kinds of programs are adapted to cardiovascular and respiratory signals:

The first concerns the processing of the raw signals to obtain the cardiac rhythm, the blood pressure and respiratory values at the onset of a cardiac peak: RR from the electrocardiogram (ECG), the systolic (SBP) and diastolic (DBP) values from the blood pressure. This processing concerns the detection of the events of interest and the resampling of the resulting signals. It will be called "**Obtention of cardiovascular and respiratory time series**".

The second concerns time, frequency and time-frequency methods for analysing these signals variabilities. It will be separated into "**Spectral analysis** " and "**Time-varying analysis**" and '**Time analysis**".

The two kinds of programs use general preprocessing methods, as filters, which will first presented.
The last section will be a practical presentation of the hierarchy of the LARY_CR directories and of the main instructions to create new programs.

2 Signal Preprocessing: Finite Impulse Response filters

2.1 Choice of the Finite Impulse Response filters

For any kind of signal processing, on raw signals as well as on resulting RR and BP series, a preprocessing is usually done to keep only events of interest for the analysis. In LARY_CR, Finite Impulse Response filters (FIR) are the most often used because the delay generated can be exactly calculated:

$$ndelay = \frac{ncoef - 1}{2} \tag{1}$$

where *ndelay* is the number of samples by which the signal is delayed and *ncoef* is the number of coefficients of the filter.

The figure 7 shows a filtering program in the SCICOS environment; the WFIR (Windowing Finite Impulse Response) process filters the raw signal X and the Signal Delay process delays X by *ndelay* samples, so raw and filtered signals remain synchronised. This point is fundamental when analysing the time relationships between two ore more signals.

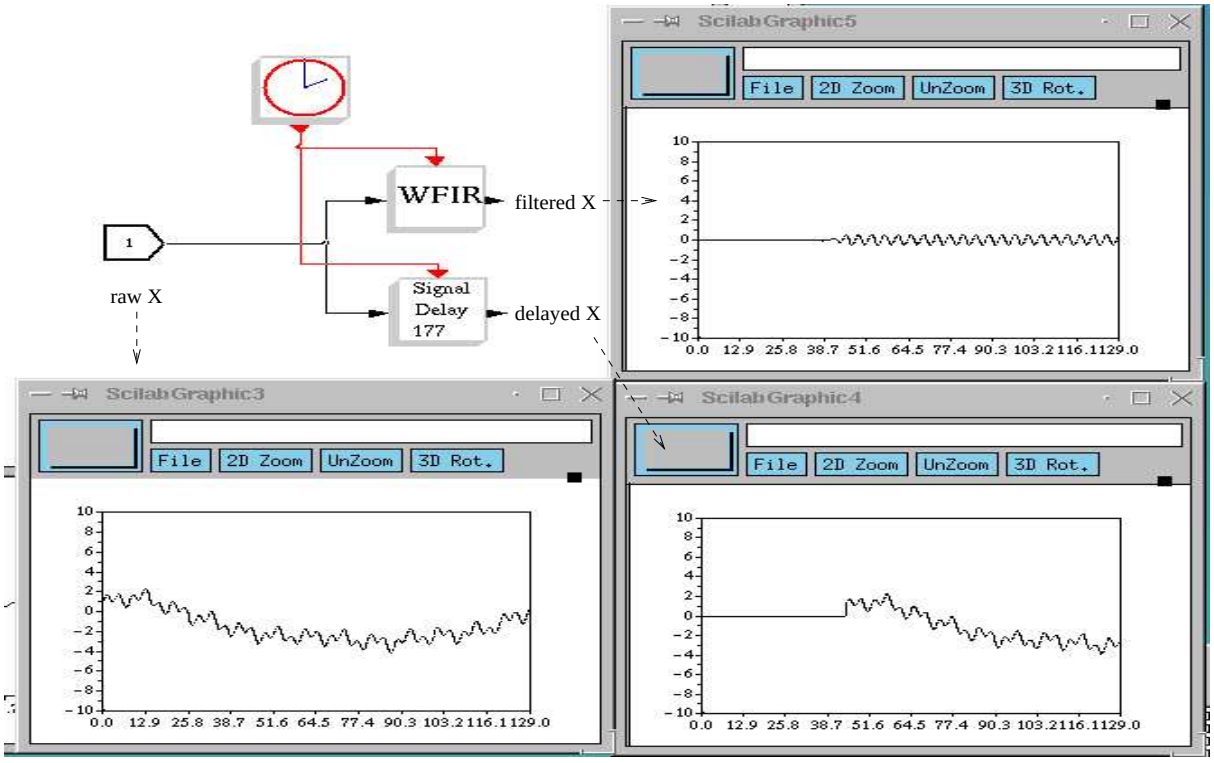


Figure 7: *General view of a Finite Impulse Response filter*

2.2 Examples of Finite Impulse Response filters, adapted to CV series

The types of WFIR used in LARY_CR are band-pass, high-pass and low-pass; they are illustrated on the next figures, using RR-like series obtained by sinusoid generators at three frequencies, close to physiological domain (0.01 Hz, 0,10 Hz, 0.25 Hz). Each figure shows the raw and filtered signals in time (in seconds) and frequency (in Hz) representations, the dialog panel and the gabarit of the chosen WFIR. The graphic representation of the gabarit is obtained each time one enters in the dialog panel; it allows the user to easely adjust the parameters of the filter according to its needs. Actual numeric values are added to the filter parameters in each case. For the high-pass and low-pass bands, only the low frequency cutoff is taken into account. The first WFIR (Fig 8) is a band-pass filter which keeps spectral activity between 0.04 and 0.20 Hz, to focuse on the 0.1 Hz. The second WFIR (Fig 9) is an high-pass filter which keeps spectral activity over 0.04 Hz, to focuse on the 0.1 Hz and the 0.25 Hz. The third WFIR (Fig 10) is a low-pass filter which keeps spectral activity below 0.04 Hz. The size of the filter (in number of points) can be changed to adjust the stiffness of the filter and in some cases, the gain is reduced, as shown on the figure 11: it is the same low-pass filter as previously described but the size of the filter is changed to 101 coefficients instead of 355 and the gain is reduced from 1 to 0.82. Correct values are restitued to the filtered signal as illustrated on the SCICOS program of the figure 11.

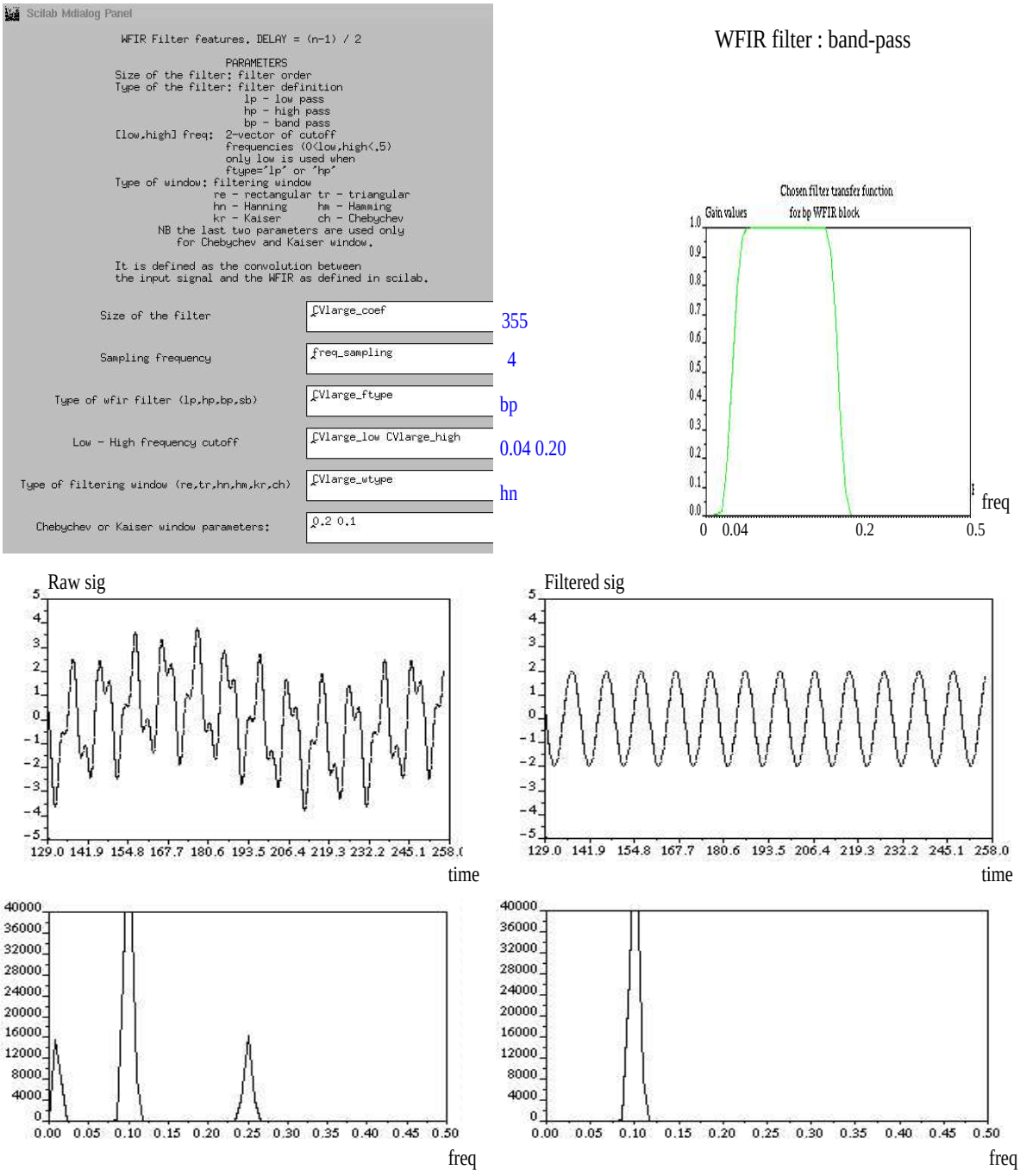


Figure 8: *Use of a pass-band Finite Impulse Response filter*

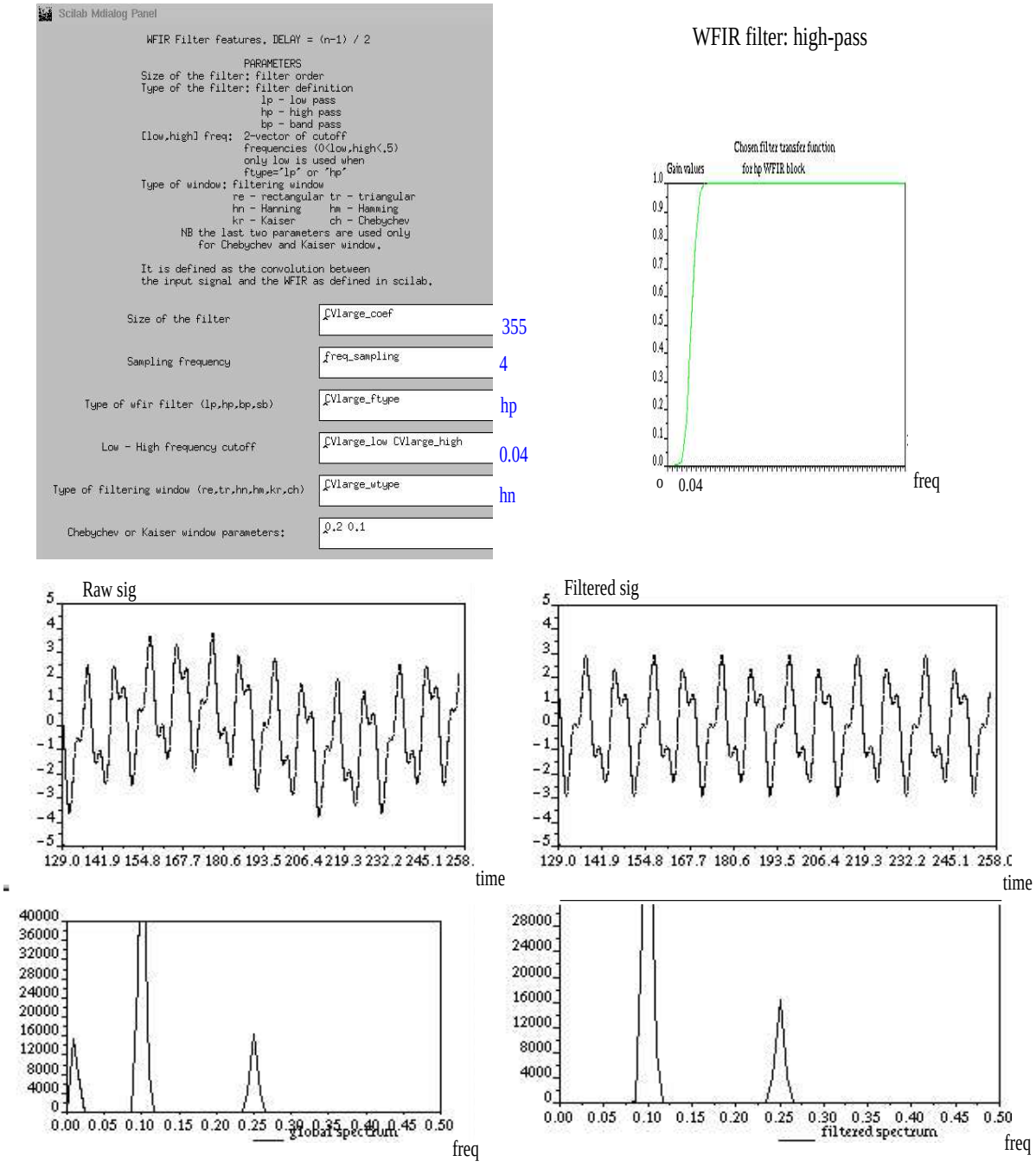


Figure 9: *Use of an high-pass Finite Impulse Response filter*

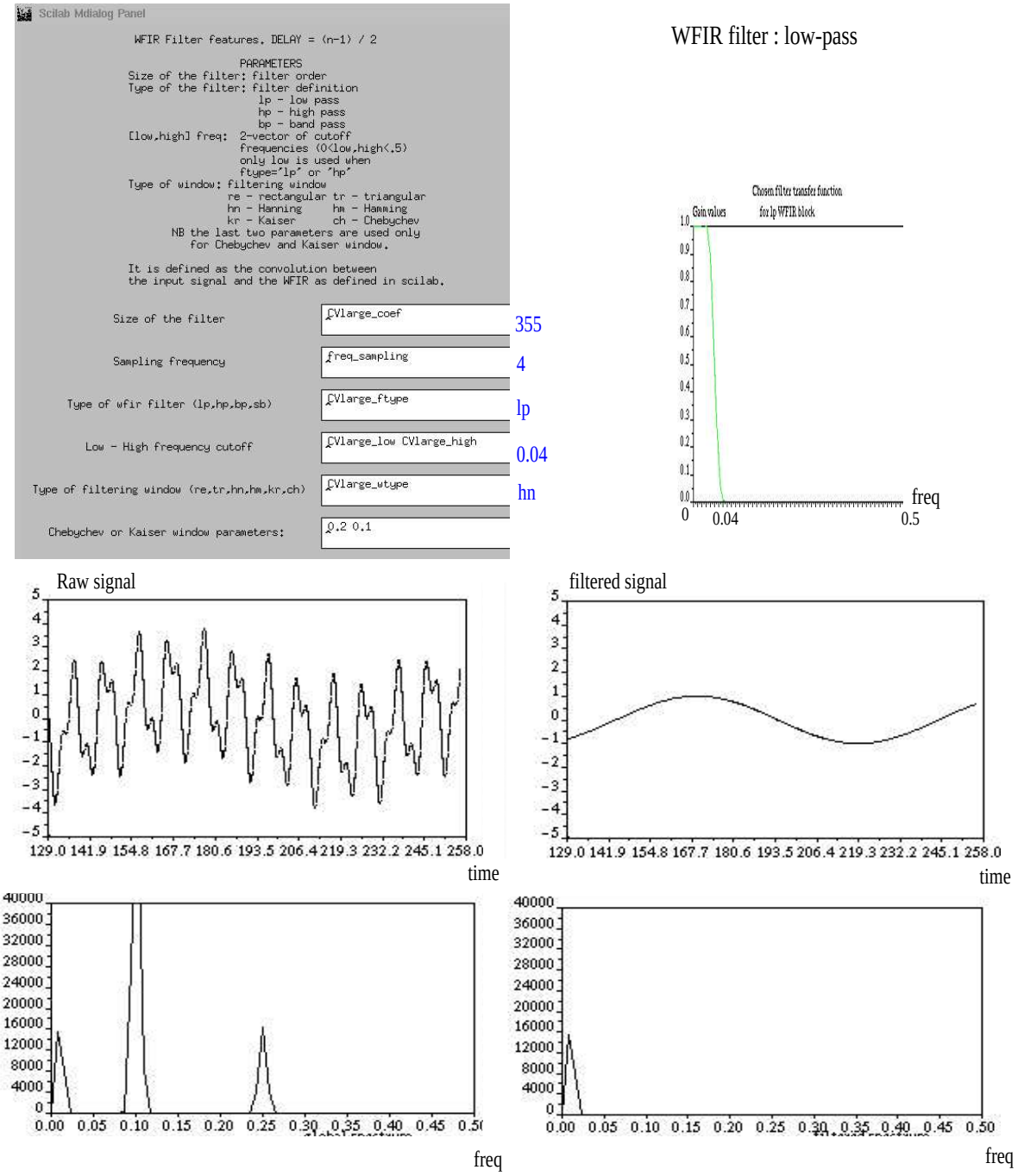


Figure 10: *Use of a low-pass Finite Impulse Response filter*

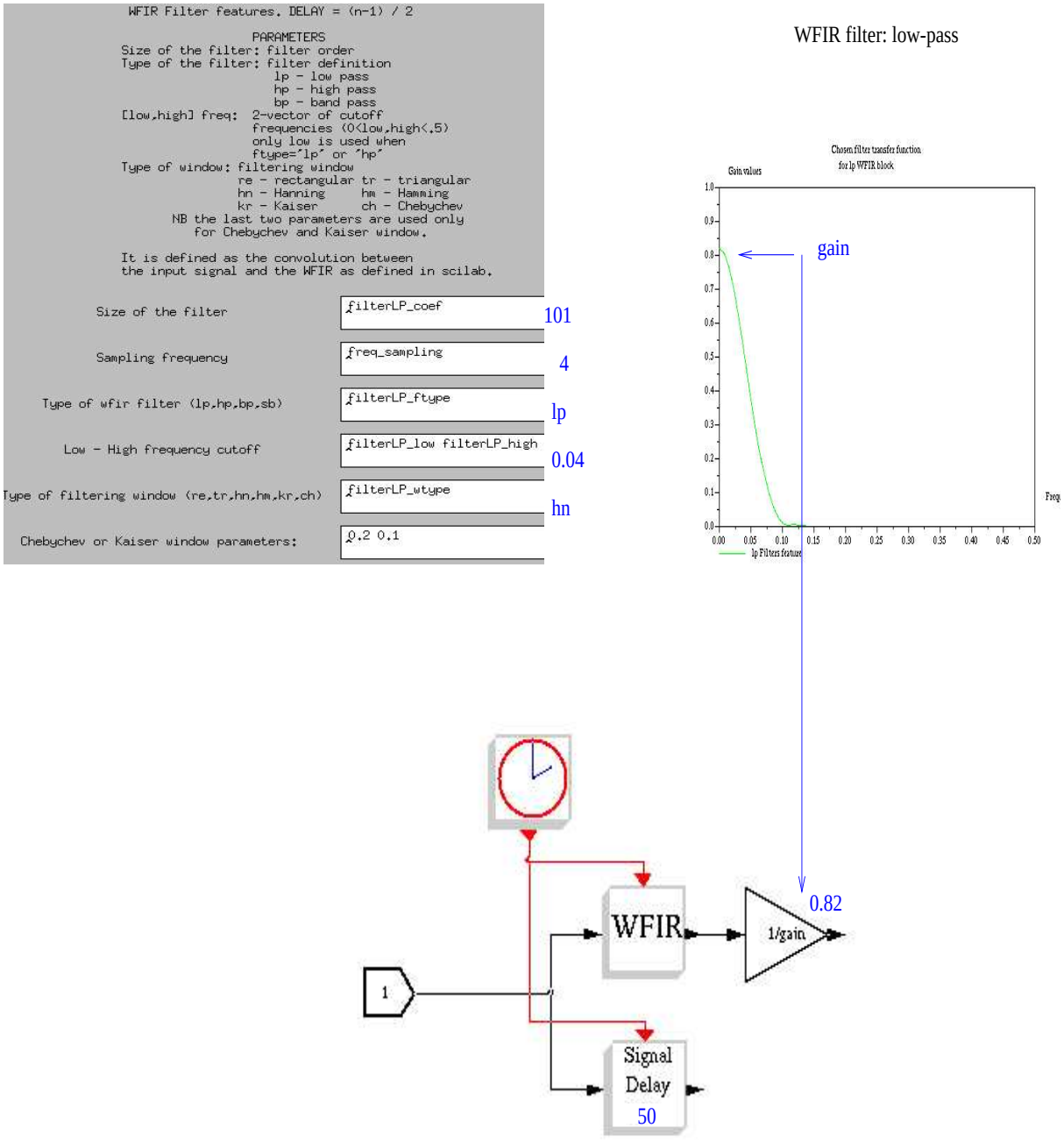


Figure 11: *Use of a low-pass Finite Impulse Response filter with adjustement of the gain*

3 Obtention of cardio-vascular and respiratory time series

3.1 Presentation

The whole flow chart processing is the following: the **Detection** program detects the events of interest from the raw cardiovascular signals; the resulting RR, SBP, DBP, PI time series are then validated by the **Visualdetection** program and resampled by spline functions in the **Resampling** program. At each step of the processing, the cardiovascular and respiratory values are accompagnied by their occurrence in time; that is a method to keep temporal relations between the different times series, and thus, to process them together and to establish relationships between them. The detection process applied to the electrocardiogram (ECG), arterial blood pressure (ABP), produces the following time series, presented in the Introduction. Each of them consists of a vector of two columns: its value and instant of detection.

- The detection process of the R peak from the ECG, carries out the RR-interval series
- The detection process of the maximum and the minimum values from the ABP, carries out respectively the systolic (SBP) and diastolic (DBP) time series and the pulse interval (PI) time series.

From the respiratory signal only the breathing frequency is estimated in the most applications. In these cases, the raw signal has no particular event to detect and is simply resampled.
The detection processes are implemented in the SCICOS graphical environment whereas the validation process of the detection and the resampling process are implemented in the SCILAB environmemt.

3.2 Detection processing

3.2.1 Principle of a detection program

The detection programs are built on the same scheme in the SCICOS environnment. The figures 12 and 13 illustrate this purpose using the RR detection processing.

The general view of the program is shown on the figure 12. Each detection block is accompanied by a *WFIR delay* block which synchronises the signals, according to the *WFIR delay* applied in the detection block (see the paragraph Preprocessing); two outputs are generated. The first output, synchronised on the reading clock, is dedicated to the raw signals and contains a time counter in seconds, the raw and the filtered signals. The second output, synchronized on the detection clock, is dedicated to the RR signal and contains the RR values and their instant of detection. The general context for the parameters is represented on the figure 14.

The view of the detection block called *RR interval detection block* on the preceding figure is now described on the figure 13. This block presents the two parallel parts of the detection processing, the first, which could be called pre-detection, to localize the QRS complex in the ECG, the second which could be called detection, to precisely detect the R peak.

- The first part of the processing, pre-detection (set of blocks 1) consists of an adaptative filter, well-suited for long time series with morphological changes. It favours the high frequency events of the cardiovascular signals, such as the QRS complex in the ECG. The

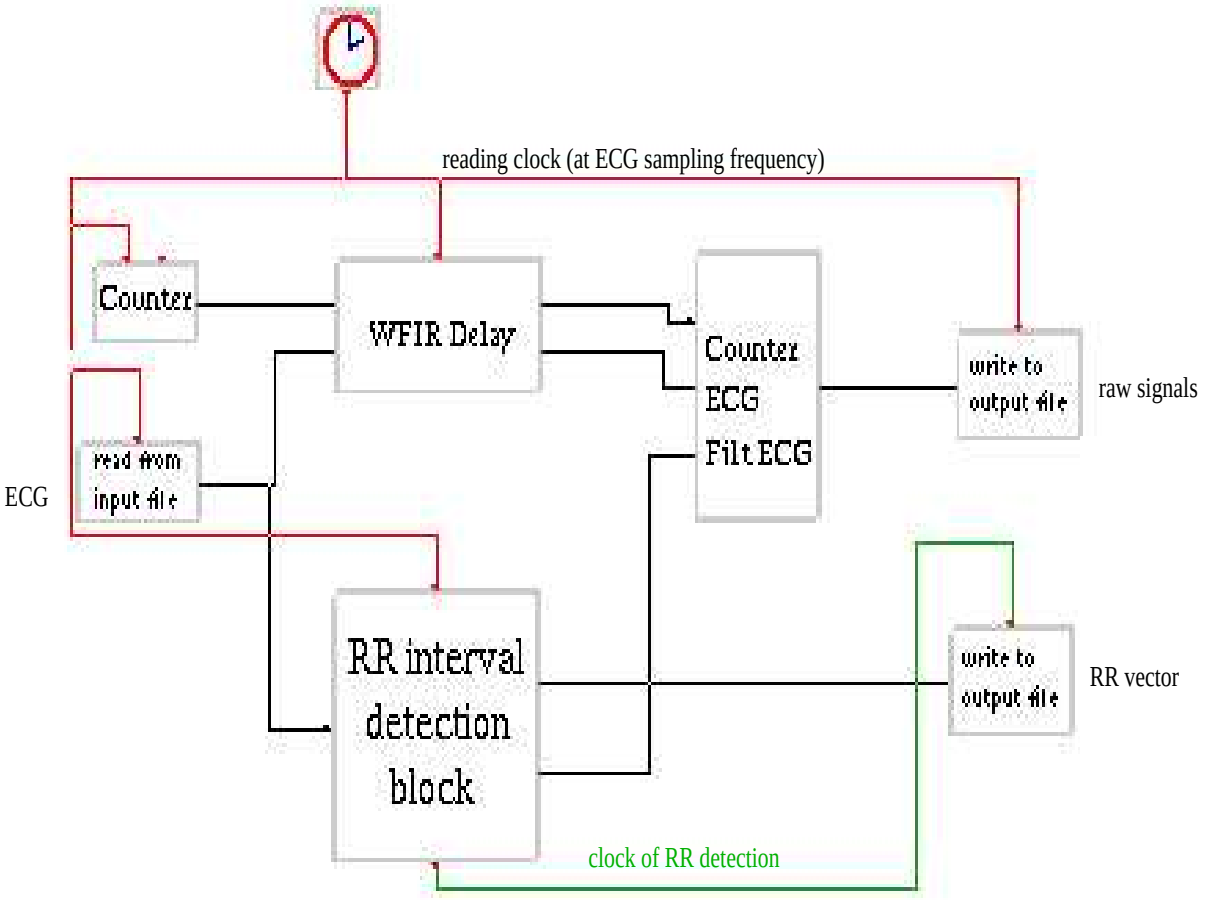


Figure 12: *General presentation of a detection program in the SCICOS environment, based here on the RR interval detection*

parameters of this adaptative filter are given on the figure 14: *FIR_ response* represents the filter coefficients and *InitResponse* is the initial response of the filter. Each time a pre-detection is true, the response of the filter is adapted with a percent of the current ECG, given by he parameter *convergence*. An adaptative threshold drives a pre-detection clock. This threshold can be adapted to the signal by the mean of the *Temps_ Constant* parameter. The graphical results are represented on the figure 13.
The pre-detection clock itself is connected with the second part of the processing and drives the precise research of the events of interest on the real signal (here the R peak on the ECG).

- This second part of the processing, detection itself (set of blocks 2) consists of a low -pass filter which smooths the raw ECG signal to allow the detection of the maximum or the minimum of the R peak. This precise detection is realized in the *RR peak Seak* block. If an extremum is really found, then a resulting vector, the value of this extremum and its instant of detection is delivered. The following paragraphs detail each of these detection algorithms. One can follow the algorithm steps during the execution, positionning the parameter *logging* on 1 and starting SCILAB as follow: scilab » DEBUG, where DEBUG is the output file of the comments.

3.2.2 RR time series

The pre-detection step can be evaluated looking at the graphical window branched on the pre-detection events: a vertical line is delivered each time a pre-detection is true (see Fig. 15).
The precise detection itself, made in the *RR peak Seak* block refers to the drawing function RRPeak4_f.sci and the simulation function rrpeak4.f. The algorithm makes a retrograd research from the instant of the pre-detection, on a window defined by *RRbuferlength*, as shown on the figure 16. The parameters are described on the figure 16, in the dialog panel and their numerical values are done in the general context, shown on the figure 14. One can choose the detection of the maximum or the minimum, using the boolean *peak*. The detection is possible under some time constraints, absolute given by *RRmin, RRmax*, relative given by *RRvariation*. The validation of this part is made later by the *VisualdetectionRR* program.

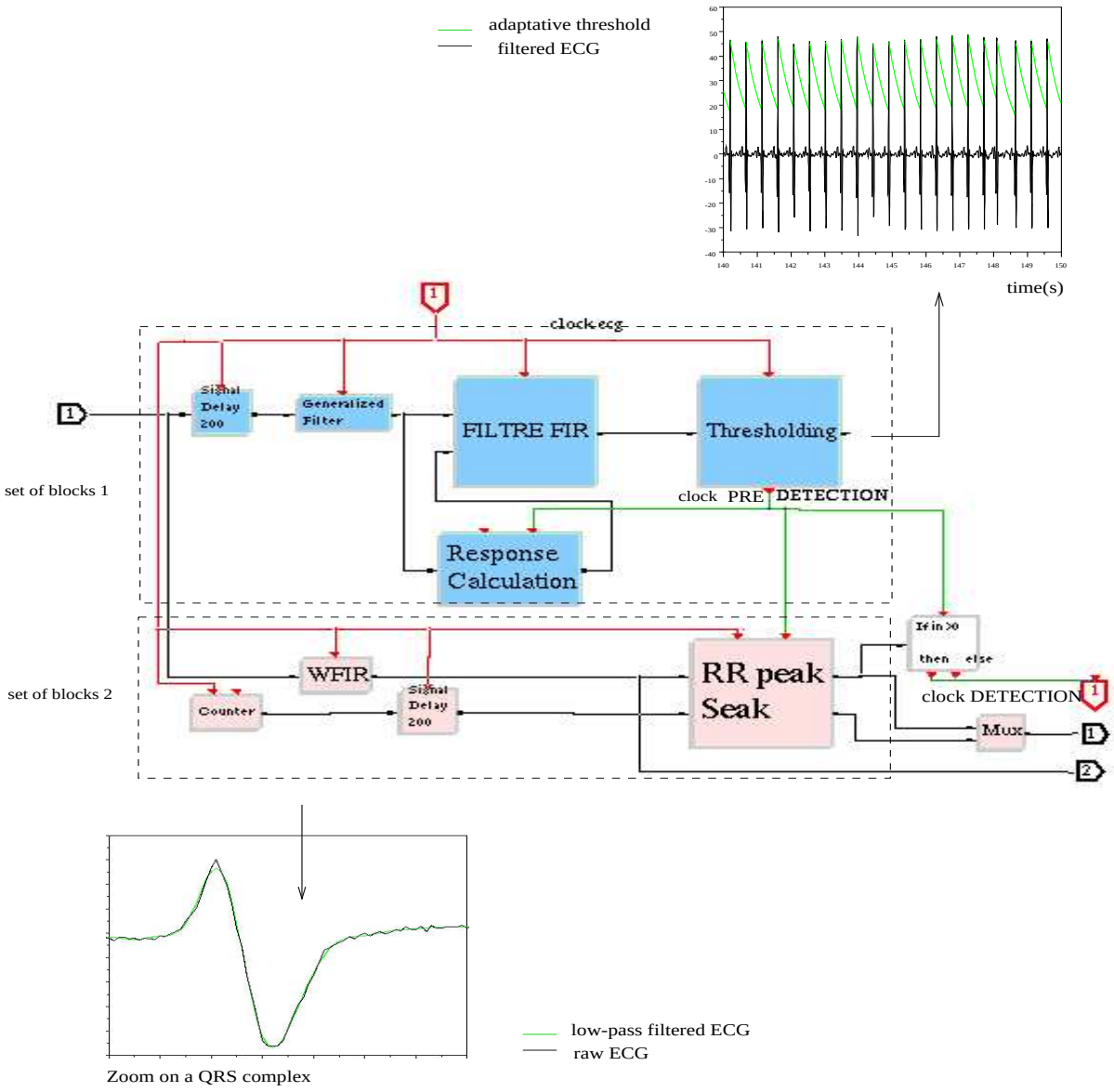


Figure 13: *The RR interval detection block*

Figure 14: *The general context of the parameters of a detection program*

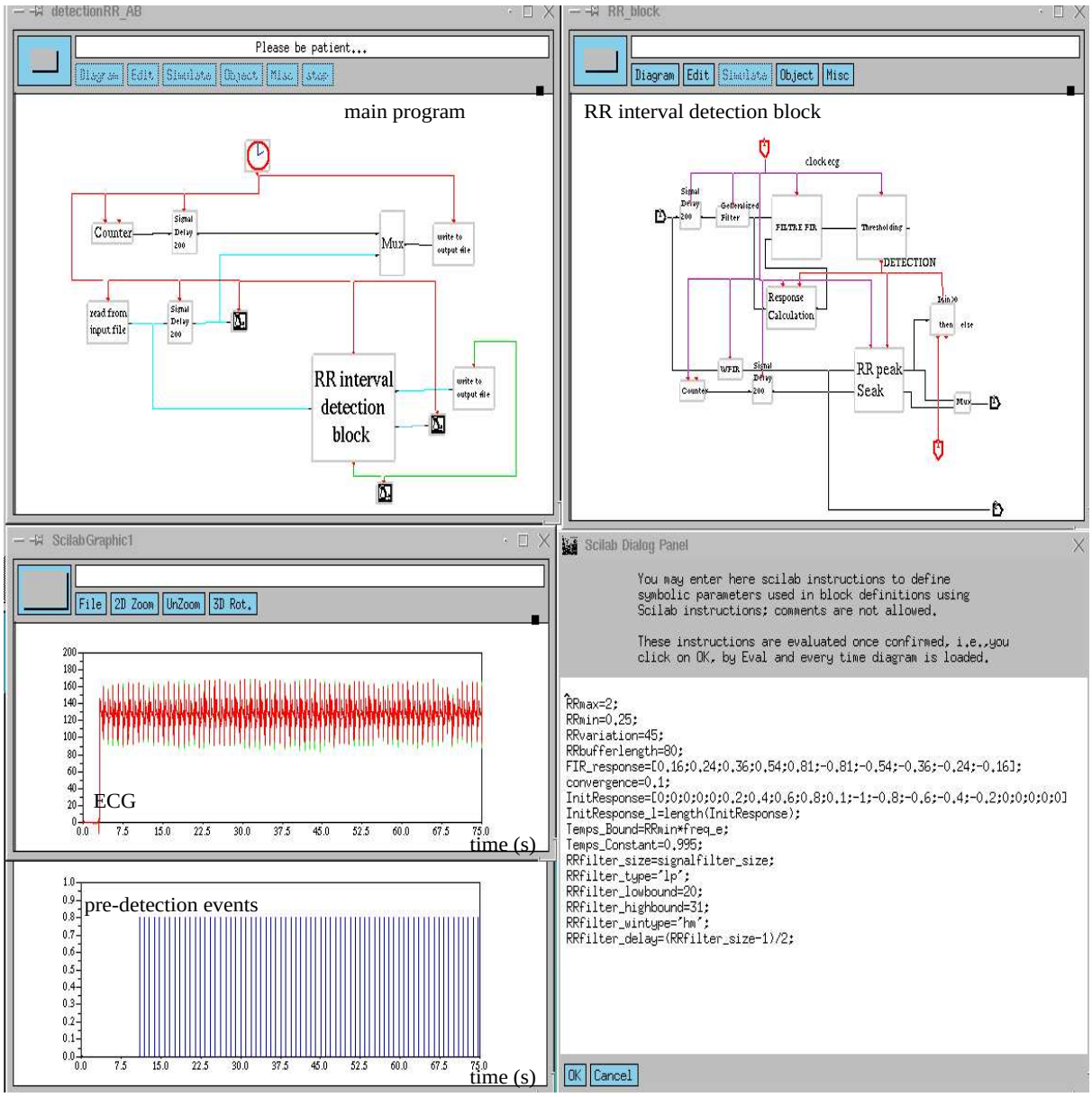


Figure 15: *R peak detection: evaluation of the pre-detection step looking at the graphical window of the pre-detection events*

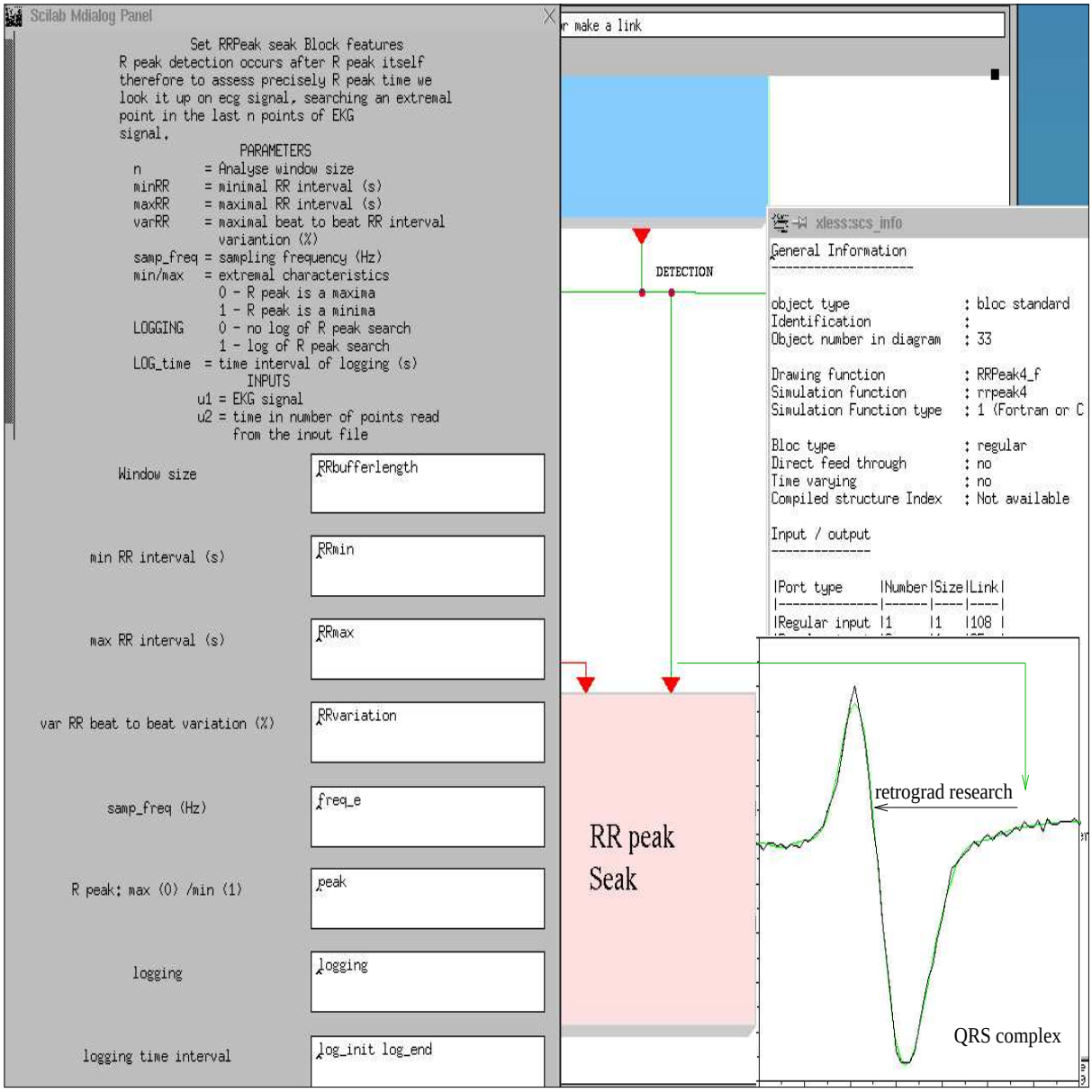


Figure 16: *The context of the RR peak Seak block*

3.2.3 Pulse Interval, Systolic and Diastolic time series

The specific BP detection programs are illustrated on the figure 18 for the main program and the figure 19 for the *Blood Pressure events detection* block. The pre-detection step can be evaluated looking at the graphical window branched on the pre-detection events: a vertical line is delivered each time a pre-detection is true. An illustration during execution is done figure 17. Some missing detections here lead to a better adjustment of the pre-detection parameters.

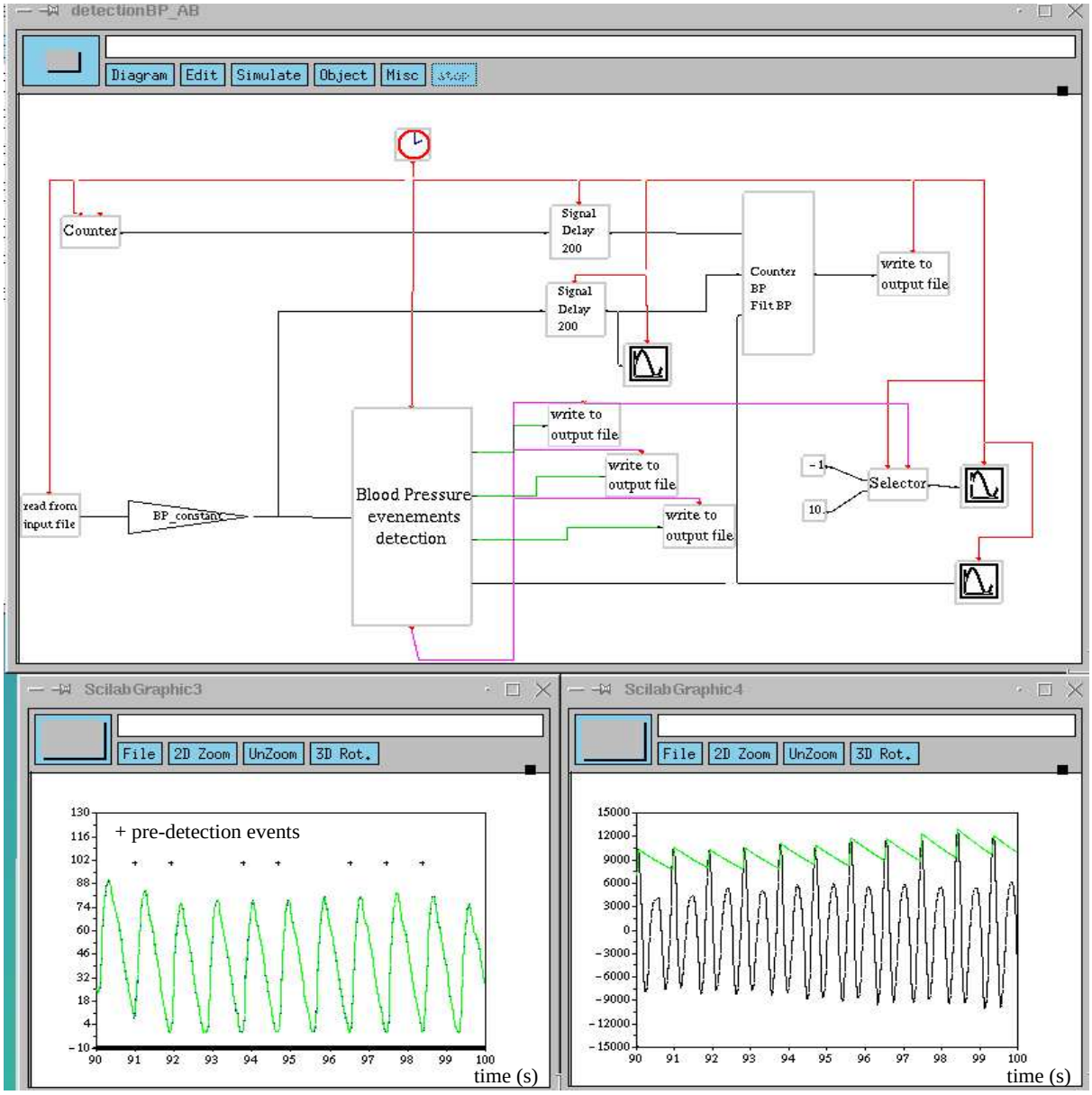


Figure 17: *SBP detection: evaluation of the pre-detection step looking at the graphical window of the pre-detection events*

The precise detection itself, made in the *PIvect SBPvect DBPvect* block, refers to the drawing function SYSDIAPeak_f.sci and the simulation function sysdiapeak.f, as shown on the figure 21. The algorithm makes a retrograd research from the instant of the pre-detection. It is described in detail at the end of this section. The parameters are described on the figure 21, in the dialog panel and their numerical values are done in the general context, shown on the figure 20. They consist of constraints in time and amplitude for SBP, DBP and PI. The validation of this part is made later by the *VisualdetectionBP* program.

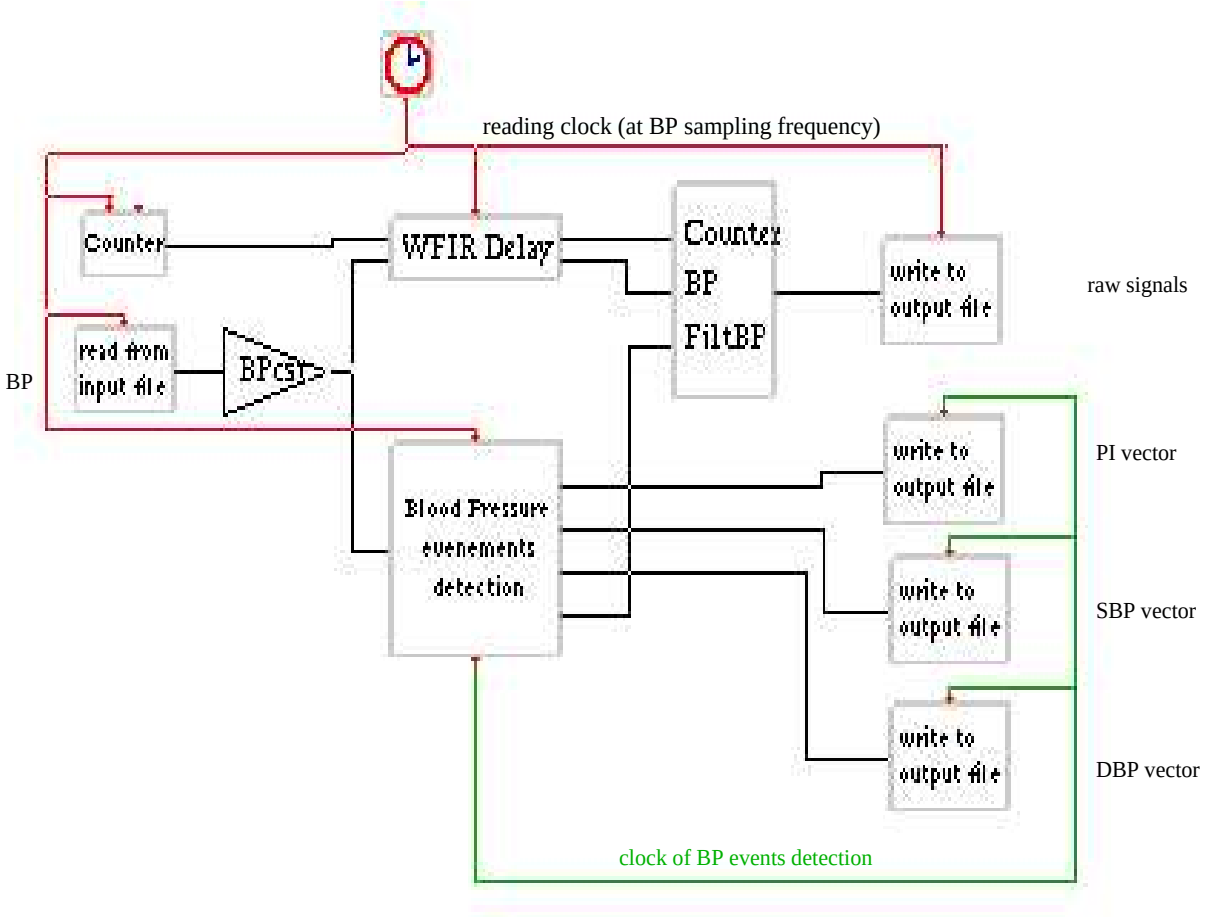


Figure 18: *The main diagram of the Blood Pressure detection*

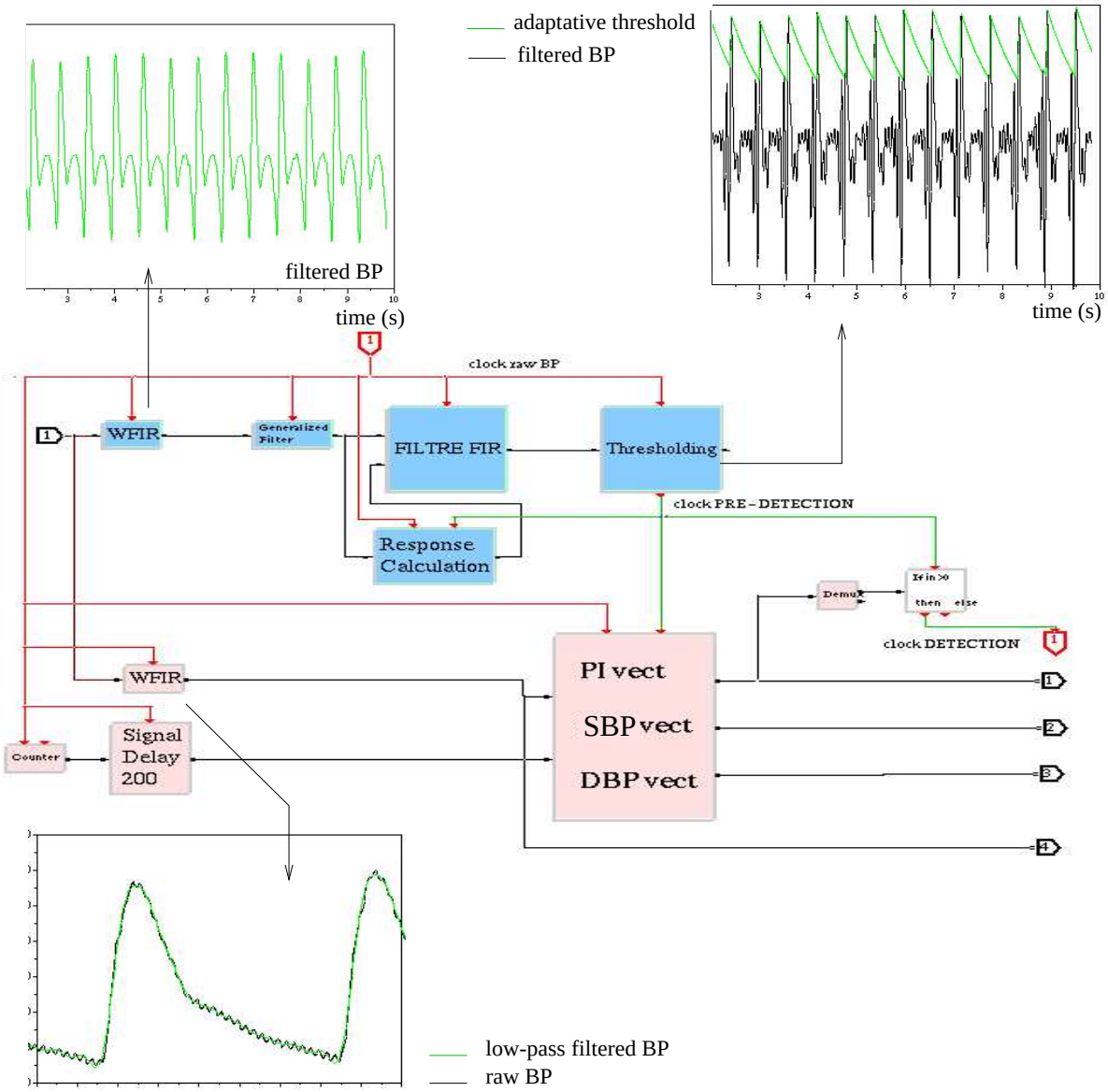


Figure 19: *The Blood Pressure events detection block*

```
freq_e=400;temps_e=1/freq_e;      sampling frequency in Hz and period in seconds
BPcst=1;                          correction factor for raw BP
filter_size=401;signaldelay=(filter_size-1)/2; size and delay for WFIR filters

Parameters of the pre-detection step (set of blocks 1):
      first filtering with a by-pass WFIR
detectionfilter_size=filter_size;
detectionfilter_type='bp';
detectionfilter_lowcut=2;
detectionfilter_highcut=15;
detectionfilter_windowtype='hm';

      next filtering with an  adaptative filter
FIR_response=[0.16;0.24;0.36;0.54;0.81;-0.81;-0.54;-0.36;-0.24;-0.16];
InitResponse=read('~/SCICOS/DETECTION/InitResponse12',-1,1);
InitResponse_l=length(InitResponse);
Temps_Bound=100;                  minimal period avoiding a new detection, in number of samples
Temps_Constant=0.997;             to modify the adaptation of the threshold
convergence=0.1;                  to modify the percent of the actual ECG in the response filter
transitoire=-200;


Parameters of the WFIR  used in the detection step (set of blocks 2):
peakfilter_size=detectionfilter_size;
peakfilter_type='lp';
peakfilter_lowcut=15;
peakfilter_highcut=180;
peakfilter_windowstype='hm';

Parameters of the "PI vector SBP vector DBP vector":
minPI=0.25;maxPI=2;PI_var=15;
minSBP=70;maxSBP=190;minDBP=40;maxDBP=110;
minSBPDBP=30;maxSBPDBP=100;mintSBPDBP=0.06;maxtSBPDBP=0.15;
BP_var=15;time_var=60;
transientperiod=5;

logging=0;log_init=1;log_end=5;
```

Figure 20: *The general context of the parameters of the Blood Pressure detection*

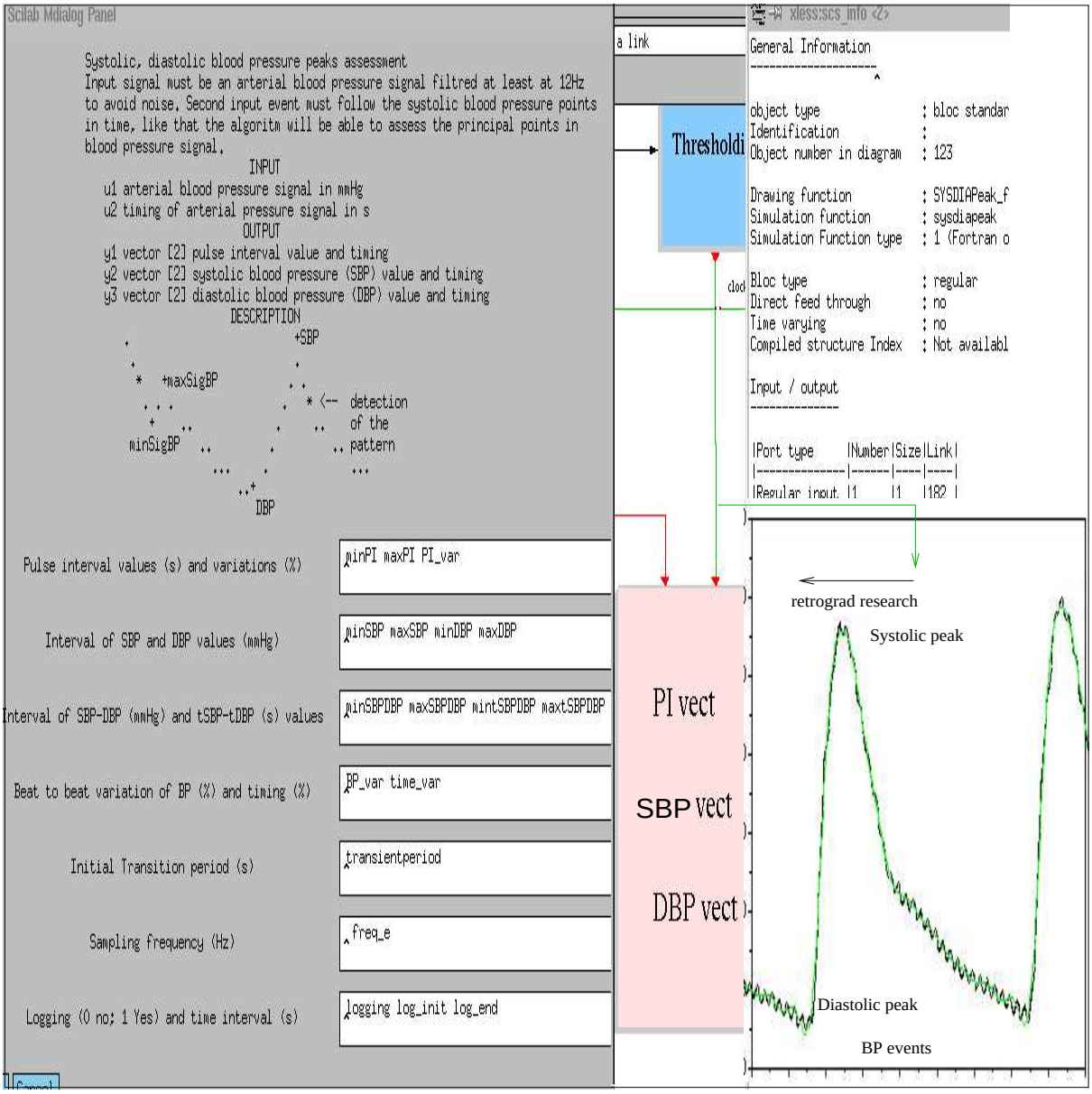


Figure 21: *The context of the PIvect SBPvect DBPvect block*

3.2.4 Respiratory signal

With the respiratory signal usually recorded by oronasal thermistor, thoracic and abdominal movement sensors, there is no event detection. However, when analysed in relationship with the cardiovascular signals, the respiratory signal has to be resynchronised with them; the synchronisation consists of adding the delay of the cardiovascular filters, as shown on the figure 22.

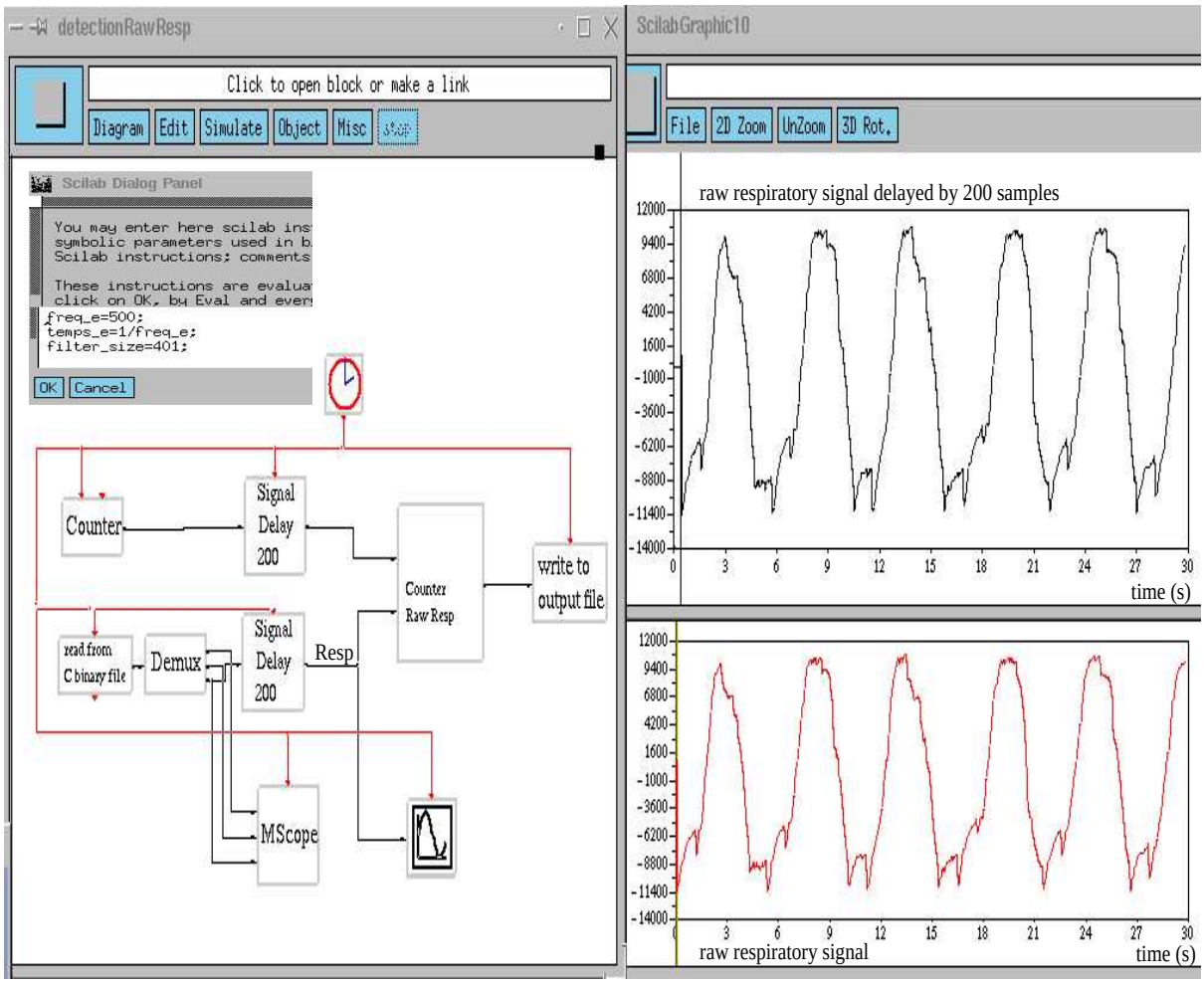


Figure 22: *Synchronisation of the respiratory signal with the cardiovascular signals*

3.3 Validation of the detection processing and resampling of the time series

3.3.1 RR time series

The three steps following the detection are shown on the figure 24. They use the SCILAB functions *RecompactRR.sci, VisualdetectionRR.sci, ResamplingRR.sci.*
The *RecompactRR* program simplifies the outputs files, removing the first column dedicated to the SCICOS simulation clock. Its input arguments are the name of the two outputs files, *RRvect* and *RawRR.*
The *VisualdetectionRR* program uses the same inputs arguments and also parameters for visualisation, expressed in seconds. This process plots the instants of the R peak detection on the raw ECG signal. This allows the user to verify the precision of the detection.
Before calling the functions in the SCILAB window, one has to verify that the inputs files are in the working directory. The *ResamplingRR* program uses the SCILAB predefined *third order spline* and *interpolation* functions. An example is done on the figure 23, applying the resampling to a sinus function. The figure 25, applied to RR time series, shows the parameters called by the resampling function, called *ResamplingRR.* The resulting resampled vector, called *RRseries* on this figure, has two columns, the RR value and the time in equidistant seconds.

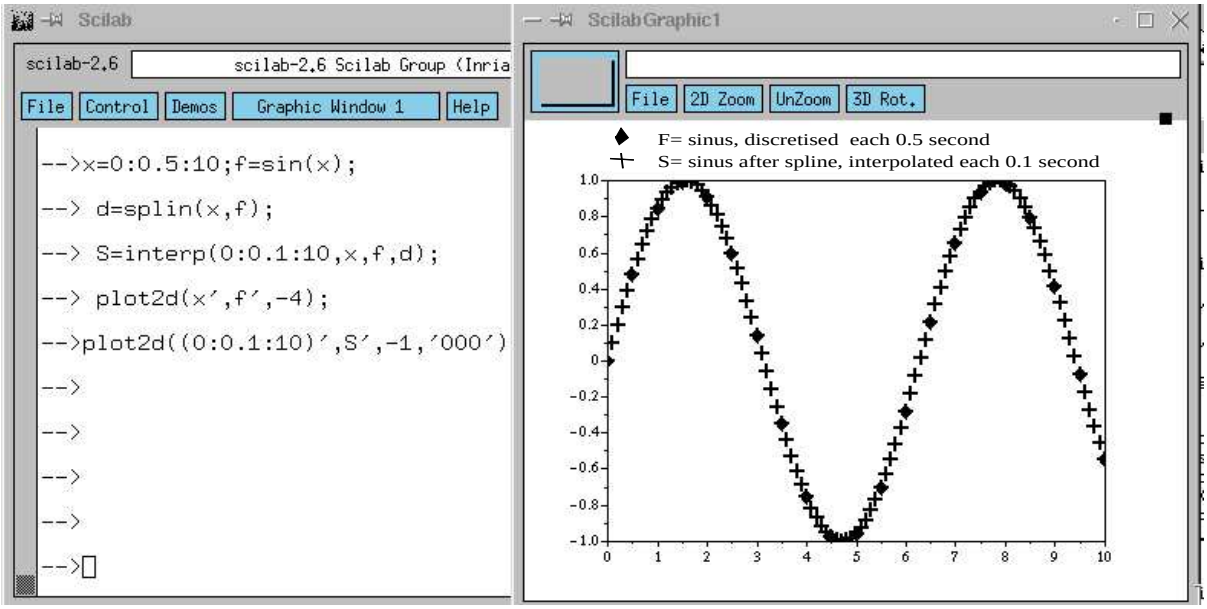
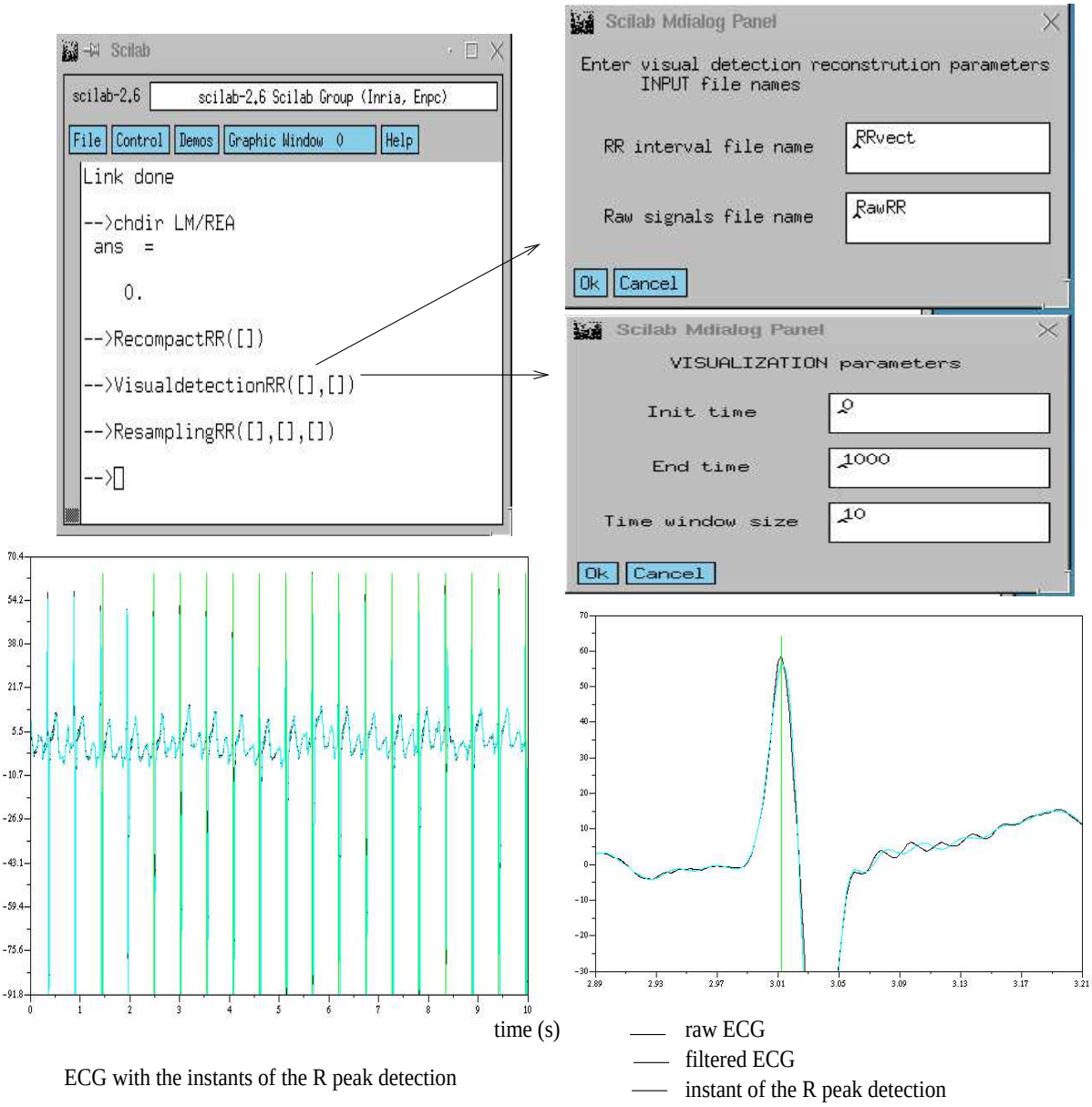


Figure 23: *Example of the resampling process applied to a sinus function*



ECG with the instants of the R peak detection

Figure 24: *The context of the VisualdetectionRR SCILAB function*

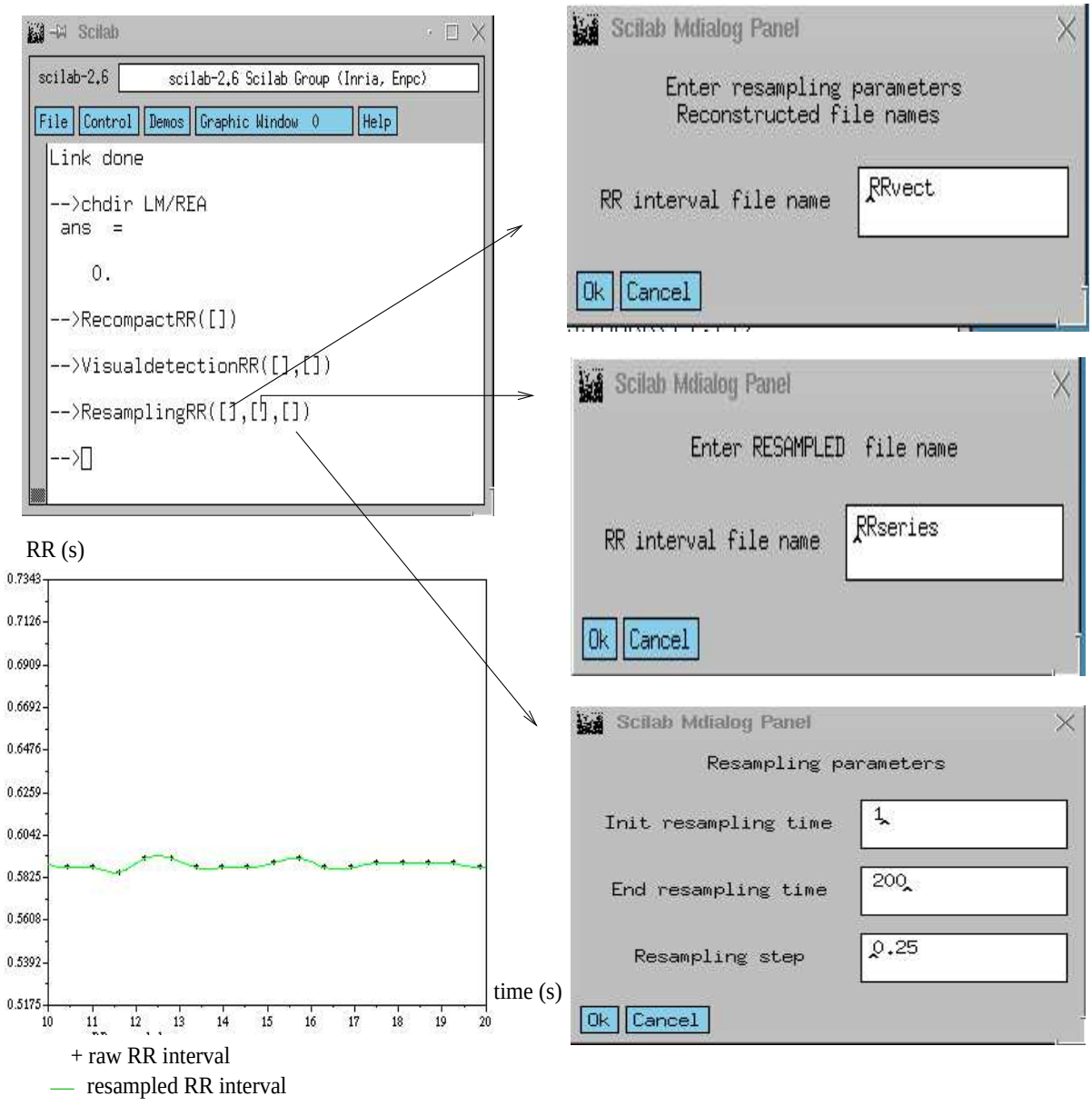
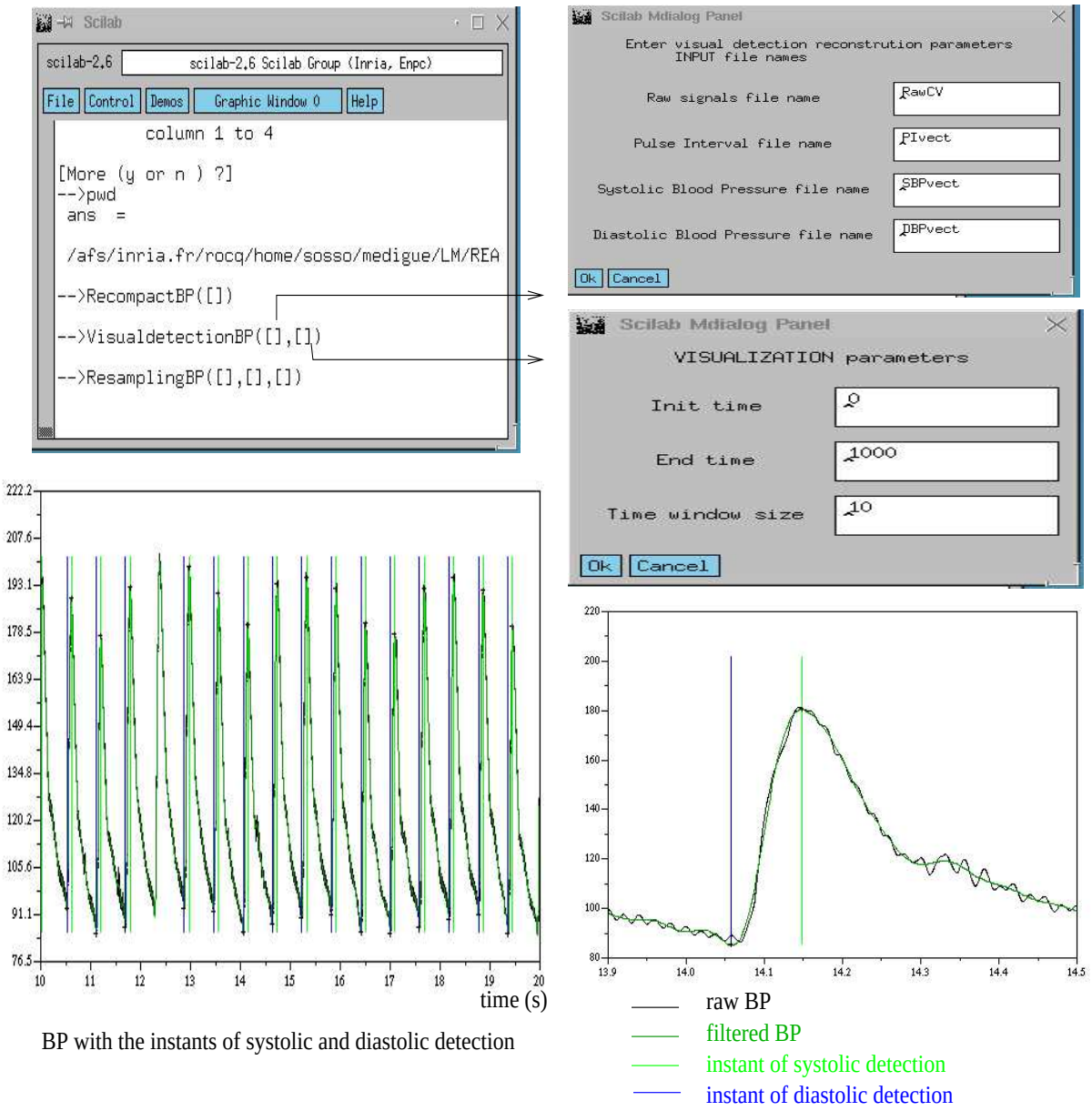


Figure 25: *The context of the ResamplingRR SCILAB function*

3.3.2 Pulse Interval, Systolic and diastolic time series

The same steps concerning the RR time series are applied to BP time series and are shown on the figure 26. The input arguments of the *RecompactBP* program are the name of the outputs files of the SCICOS detection program, *SBPvect*, *DBPvectPIvect* and *RawCV*.
The *VisualdetectionBP* program uses the same inputs arguments and also parameters for visualisation, expressed in seconds. This process plots the instants of the extremums detection, systolic and diastolic, on the raw BP signal. The figure 27 shows the parameters called by the resampling process.



BP with the instants of systolic and diastolic detection

Figure 26: *The context of the VisualdetectionBP SCILAB function*

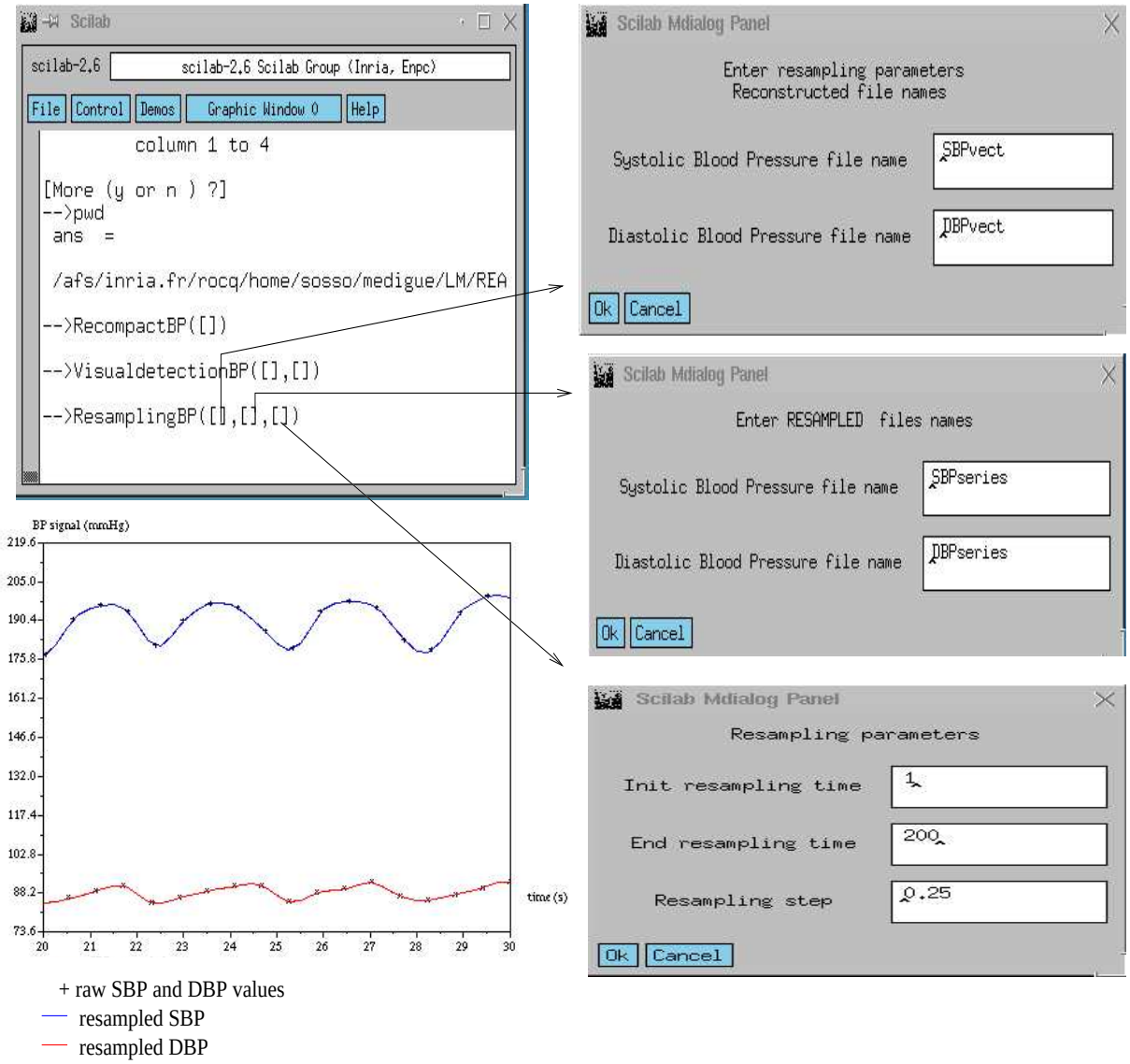


Figure 27: *The context of the ResamplingBP SCILAB function*

3.3.3 Respiratory signal

The raw respiratory signal is simply resampled at the same frequency as the cardiovascular time series. This is done by the *ResamplingRawResp.sci* function, which is applied after the *RecompactRawResp.sci* function, as shown on the figure 28.

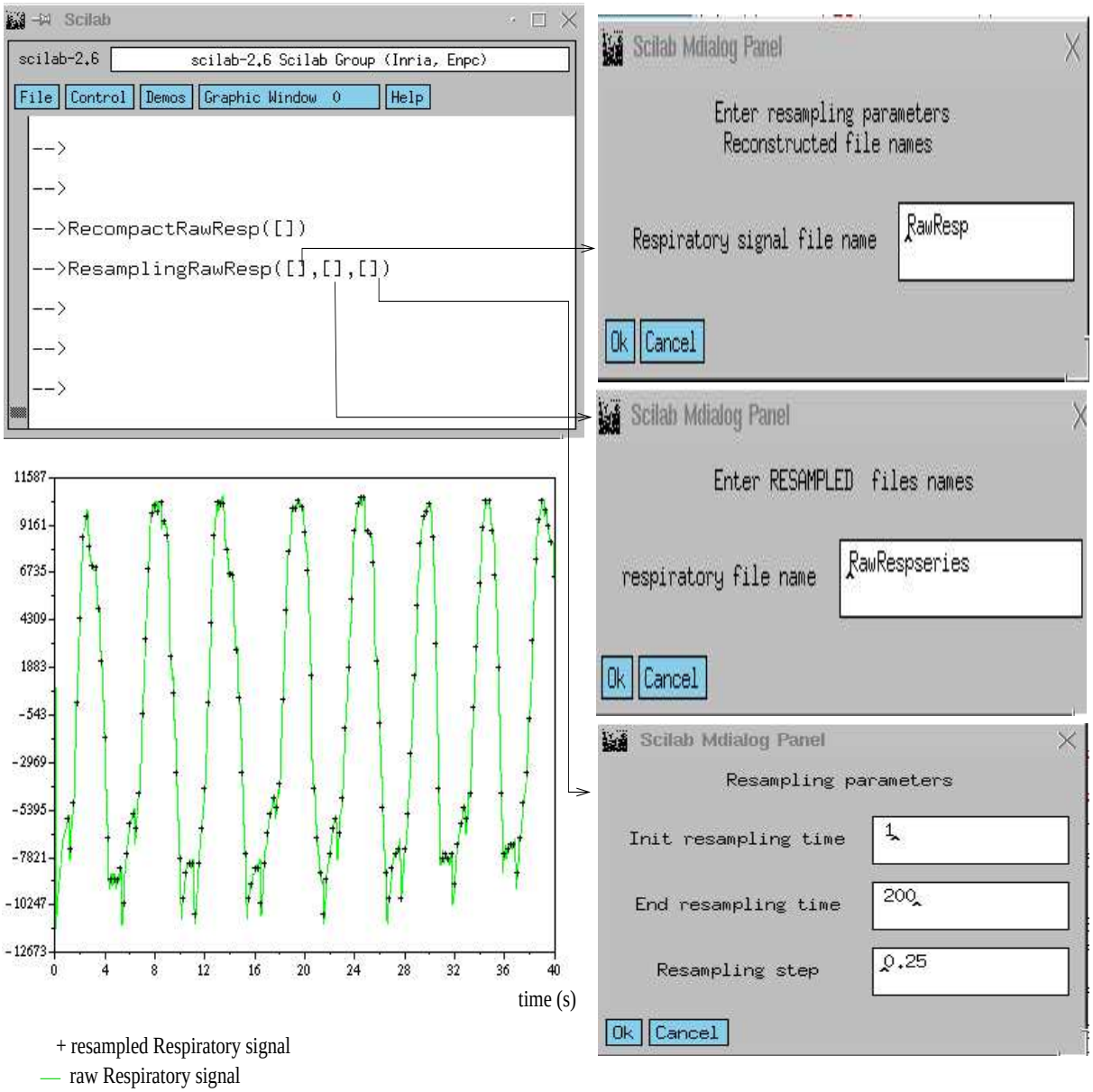


Figure 28: *The context of the ResamplingRawResp SCILAB function*

3.4 Description of the systolic and diastolic research algorithm

3.4.1 The *sysdiapeak.f* function

The simulation function sysdiapeak.f, written in Fortran, allows the research of the extremums of the blood pressure, and supplies vectors SBPvect and DBPvect, corresponding respectively to the systolic and diastolic pressures, and the vector PIvect, corresponding to the time interval between two maximums. The signal treated by sysdiapeak.f must be completely of the same sign (positive or negative). Principle of the research of extremums:The algorithm works in a retrograde way: from the instant of pre-detection corresponding to a point of the curve of blood pressure, provided with an index, sysdiapeak.f looks for the absolute maximum, then the absolute minimum, preceding the point of initiation of the research. Parameters of control (cSBP, with SBP the maximum, cDBP, with DBP the minimum, cPI, with PI the time interval between two maximums...), functions of the last one detected extremum, allow, after comparison to the recently obtained values, to validate or to reject these ones.The research of extremums begins with that of the first absolute maximum (SYS). The conditions which must obey a point z(index) to be validated as maximum are z(index) $>$ cSBP(1), z(index) $<$ cSBP(2) and maxp=1; cSBP(1) and cSBP(2) are respectively the lower and superior borders of control of the maximum; maxp is the result of the function maxpoint, looking for the maximum point, and maxp is equal to 1 if z(index) is a maximum. If these conditions are verified and that the index of the point in question is superior to 4, that is the point is not situated in the left limit of the window, which is at present analyzed, then one allocates to the variable SYS the value z(index), corresponding to the amplitude of the detected maximum, and to the variable tSYS the value Buffert-ind, corresponding to an index, which divided by the frequency freq_s, supplies us the time of detection of the maximum. If the conditions are not verified, index is decremented and ind is incremented, and the new z(index) is subjected to the previously described conditions, after one has applied on it the maxpoint function, et cetera until the validation of a maximum. On the other hand, if index is lower than 4, the detection of the systolic pressure failed and one attributes the value zero to variables SYS and tSYS.In a second time, one looks for the minimum preceding the previously detected maximum. The principle of detection of a minimum is identical to what was explained higher about the detection of a maximum, cSBP(1), cSBP(2), and maxp being replaced here by cDBP(1), cDBP(2) and minp; furthermore, the minpoint function looks for the minimum point. When the three conditions defining a minimum are verified and when index $>$ 4, one attributes to the variable DIA the value z(index) and to the variable tDIA the value Buffert-ind. Some indexes allow to delimit the zone of research of the minimum: these indexes are index_startDBP and index_stopDBP; they depend of ctSBPDBP(1) and ctSBPDBP(2), which are both borders of control of the time interval tSBP-tDBP. The situation where index $<$ index_stopDBP is envisaged: in that case, the previously detected maximum is not followed by a minimum in the good interval; one starts again from index_SYS-1 and ind_SYS+1, situated after the indexes characterizing the last detected maximum, and the first part of the algorithm is again executed from 10, so as to find a new maximum. This is also made when the cDBP(1) $>$ cDBP(2) condition is verified, this corresponding in the failure of the research of minimum, resulting from an incorrect maximum found in the first part of sysdiapeak.f. Besides the research of minimum, this part of the algorithm can detect a maximum, which is superior to that found in the first part, this being only a relative maximum; in that case, SYS, tSYS, and the indexes characterizing the discovered maximal point (index_SYS and ind_SYS) are updated and the execution of the second part of sysdiapeak.f must begin again. So, sysdiapeak.f supplies the

maximum of the relative maximums. In the third time, the algorithm looks for a new maximum preceding the minimum, which has been detected during the second stage, so as to be sure that the signal of pressure is periodic. The coordinates of this maximum are attributed to SYS2 and tSYS2 (index). Furthermore, this part can also discover a minimum, which is lower than that found in the previous part, and the corresponding index of which is situated between index_startDBP and index_stopDBP; DIA and tDIA are then updated and the third part of sysdiapeak.f is again executed. So, sysdiapeak.f supplies the minimum of the relative minimums. In the fourth time, sysdiapeak.f looks for the minimal value preceding the last detected maximum, so as to verify the rightness of SYS2. The coordinates of this minimum are attributed to DIA2 and tDIA2. The index_startDBP2 and index_stopDBP2 indexes delimit the zone where the minimum is susceptible to be present. In 370, the algorithm presents a block allowing to detect any minimum lower than the one which was previously found, and this as long as index $>$ index_stopDBP2 and index > 4; in the case of the detection of a new minimum, DIA2 and tDIA2 are updated.The cases index $<$ index_stopDBP2 and cDBP(1) $>$ cDBP(2) are treated: in a similar way as what was said concerning the second part, one starts again from index_SYS2-1 and ind_SYS2+1, and the third part of the algorithm is again executed from 310. This part also allows to find a maximum whose amplitude z(index) is superior to SYS2. The end of sysdiapeak.f presents in 110 the affectations of the values resulting from detections made over a period of the blood pressure signal. y1(1)...y3(2) variables correspond to the 3 output vectors, and z(n+1)...z(n+6) variables represent the state of detection. Two cases appear: either bool_v=0, what corresponds to a successful detection, and variables take the values recently obtained by detection, or bool_v=1, what corresponds to a failure of detection, and variables take the value zero, except z(n+6), so as to avoid an infinite test.

3.4.2 La fonction *sysdiapeak.f*

La fonction de simulation sysdiapeak.f, écrite en Fortran, permet la recherche des extrêmums de la pression artérielle, et fournit les vecteurs SBPvect et DBPvect, correspondant respectivement aux pressions systoliques et diastoliques, ainsi que le vecteur PIvect, correspondant à lintervalle de temps entre deux maximums. Le signal traité par sysdiapeak.f doit être entièrement du même signe (positif ou négatif). Principe de la recherche des extrêmums : Lalgorithme fonctionne de manière rétrograde : à partir de linstant de pré-détection correspondant à un point de la courbe de pression artérielle, muni dun indice, sysdiapeak.f recherche le maximum absolu puis le minimum absolu précédant temporellement le point d'initiation de la recherche. Des paramètres de contrôle (cSBP, avec SBP le maximum, cDBP, avec DBP le minimum, cPI, avec PI l'intervalle de temps entre deux maximums), fonctions du dernier extrêmum détecté, permettent, après comparaison aux valeurs nouvellement obtenues, de valider ou de rejeter ces dernières.La recherche des extrêmums débute par celle du premier maximum absolu (SYS). Les conditions auxquelles doit obéir un point z(index) pour être mémorisé en tant que maximum sont z(index)$>$cSBP(1), z(index)$<$cSBP(2) et maxp=1 ; cSBP(1) et cSBP(2) sont respectivement les bornes inférieure et supérieure de contrôle du maximum ; maxp est le résultat de la fonction maxpoint, recherchant le point maximum et maxp est égal à 1 si z(index) est un maximum. Si ces conditions sont vérifiées et que l'index du point en question est supérieur à 4, c'est à dire le point n'est pas situé à la limite gauche de la fenêtre actuellement analysée, alors on affecte à la variable SYS la valeur z(index), correspondant à l'amplitude du maximum détecté, et à la variable tSYS la valeur Buffert-ind, correspondant à un indice, qui divisé par la fréquence freq_s, nous fournit

le temps de détection du maximum.Si les conditions ne sont pas vérifiées, index est décrémenté et ind est incrémenté, et le nouveau z(index) est soumis aux conditions précédemment décrites, après être passé dans la fonction maxpoint, et ainsi de suite jusqu'à la validation d'un maximum. Par contre, si index est inférieur à 4, la détection de la pression systolique a échoué et on attribue la valeur 0 aux variables SYS et tSYS.Dans un second temps, on recherche le minimum précédant temporellement le maximum détecté auparavant. Le principe de détection d'un minimum est identique à ce qui a été exposé plus haut au sujet de la détection d'un maximum, cSBP(1), cSBP(2), et maxp étant ici remplacés par cDBP(1), cDBP(2) et minp ; de plus, la fonction minpoint recherche le point minimum. Lorsque les trois conditions définissant un minimum sont vérifiées et que index>4, on attribue à la variable DIA la valeur z(index) et à la variable tDIA la valeur Buffert-ind.Des indices permettent de délimiter la zone de recherche du minimum : il s'agit de index_startDBP et index_stopDBP ; ces indices dépendent de ctSBPDBP(1) et ctSBPDBP(2), qui sont les deux bornes de contrôle de l'intervalle de temps tSBP-tDBP. La situation où index<index_stopDBP est envisagée : dans ce cas, le maximum détecté précédemment n'est pas suivi par un minimum dans le bon intervalle ; on repart alors des indices index_SYS-1 et ind_SYS+1, situés après les indices caractérisant le dernier maximum détecté, et la première partie de l'algorithme est exécutée à nouveau à partir de 10, de façon à trouver un nouveau maximum. Ce retour en arrière est également effectué lorsque la condition cDBP(1)>cDBP(2) est vérifiée, ceci correspondant à l'échec de la recherche de minimum, provenant d'un maximum incorrect trouvé dans la première partie de sysdiapeak.f.Outre la recherche de minimum, cette partie de l'algorithme peut détecter un maximum supérieur à celui mis en évidence dans la première partie, ce dernier n'étant alors qu'un maximum relatif ; dans ce cas, SYS, tSYS, ainsi que les indices caractérisant le point maximal détecté (index_SYS et ind_SYS) sont remis à jour et l'exécution de la seconde partie de sysdiapeak.f doit recommencer. Ainsi, sysdiapeak.f fournit le maximum des maximums relatifs. Dans le troisième temps, l'algorithme recherche un nouveau maximum précédant, au niveau temporel, le minimum détecté au cours de la seconde étape, de façon à être sûr que le signal de pression est périodique. Les coordonnées de ce maximum sont affectées à SYS2 et tSYS2 (indice).De plus, cette partie peut également détecter un minimum inférieur à celui trouvé dans la partie précédente, et dont l'indice correspondant est situé entre index_startDBP et index_stopDBP ; DIA et tDIA sont alors remis à jour et la troisième partie de sysdiapeak.f est exécutée une nouvelle fois. Ainsi, sysdiapeak.f fournit le minimum des minimums relatifs.Dans le quatrième temps, sysdiapeak.f recherche la valeur minimale précédant le dernier maximum détecté, de manière à vérifier l'exactitude de SYS2. Les coordonnées de ce minimum sont attribuées à DIA2 et tDIA2. Les indices index_startDBP2 et index_stopDBP2 délimitent la zone où le minimum est susceptible d'être présent. En 370, l'algorithme présente un bloc permettant de détecter tout minimum inférieur à celui qui a été trouvé précédemment, et ceci tant que index>index_stopDBP2 et index>4 ; dans le cas de la détection d'un nouveau minimum, DIA2 et tDIA2 sont remis à jour.Les cas index<index_stopDBP2 et cDBP(1)>cDBP(2) sont traités : de manière similaire à ce qui a été dit concernant la seconde partie, on repart alors des indices index_SYS2-1 et ind_SYS2+1, et la troisième partie de l'algorithme est exécutée à nouveau à partir de 310. Cette partie permet également de mettre en évidence un maximum d'amplitude z(index) supérieure à SYS2. La fin de sysdiapeak.f présente au niveau de 110 les affectations des valeurs résultant des détections effectuées sur une période du signal pression artérielle. Les variables y1(1)y3(2) correspondent aux 3 vecteurs de sortie, et les variables z(n+1)z(n+6) représentent l"etat de détection. Deux cas figurent : soit bool_v=0, ce qui correspond à une détection réussie, et les variables prennent les valeurs nouvellement obtenues par

détection, soit bool_v=1, ce qui correspond à un échec de détection, et les variables prennent une valeur nulle, sauf z(n+6) de façon à éviter un test infini.

4 Spectral analysis of cardiovascular and respiratory time series

4.1 Presentation

In the spectral domain, we are used to analyse separately the cardiovascular and respiratory time series with the Fast Fourier Transform (FFT), which gives their power spectral Density.
We are used also to analyse the cardiovascular and respiratory time series two by two, by studying their Coherence and Gain. In particular, the ratio of the RR-interval to SBP variability evaluates the spectral Baroreflex Sensitivity (BRS).
From the whole power spectrum density, as well as from the whole spectral Coherence and Gain, we extract the two bands of interest described in the physiological introduction.

4.2 Fast Fourier Transform

4.2.1 Introduction to the frequential smoothing

To improve the FFT results, we are used to apply filters before and after the FFT process itself, as shown on the figure 32. Before, the operatins on the scalar signal, *raw X*, can be the substraction of the mean of the signal and the WFIR processing, as already described in the **Preprocessing** section. After, the vectorial signal is smoothed to obtain the Smooth Power Spectral Density (SPSD).
The figures 29, 30, 31 show the graphical effects of the spectrum smoothing and the importance of the choice of an appropriate window. The smoothing features automatically happen when clicking on the *smooth* block.

The figures 29 and 32 refer to the mathematical definitions given in the Research Report.

4.2.2 Smoothed Power Spectral Density assessment in the whole spectrum

The three following figures 32, 33 and 34 illustrate the global processing to obtain the SPSD: the first shows the SCICOS diagram, the second, the parameters of each process block and the third, the graphical successive features of the spectrum.

4.2.3 SPSD and spectral Dispersion assessment in a frequency band

The variables computed in each frequency band, the energy and dispersion, completely described the spectrum. The dispersion is evaluated by three values: the frequency around wich the spectral dispersion is minimal, the SPSD for this minimal dispersion and the dispersion indice of the spectrum. Each of the following figures described the parameters for the *Power band* and the *Dispersion Band* blocks.
Three possibilities are given: for RR and BP variabilities analysis in low frequency, we analyse a fixed band; in high frequency, we analyse a fixed band when the respiratory signal is not recorded or when its frequency is too much variable; but when we have a stable respiratory signal, we can analyse a band centered around its central frequency; this situation is useful for paced breathing rate, because it allows to choice a narrow band around the center frequency of the respiration. In this last case, the *Power Band* block is replaced by the *Power center* block, which uses the central respiratory frequency.

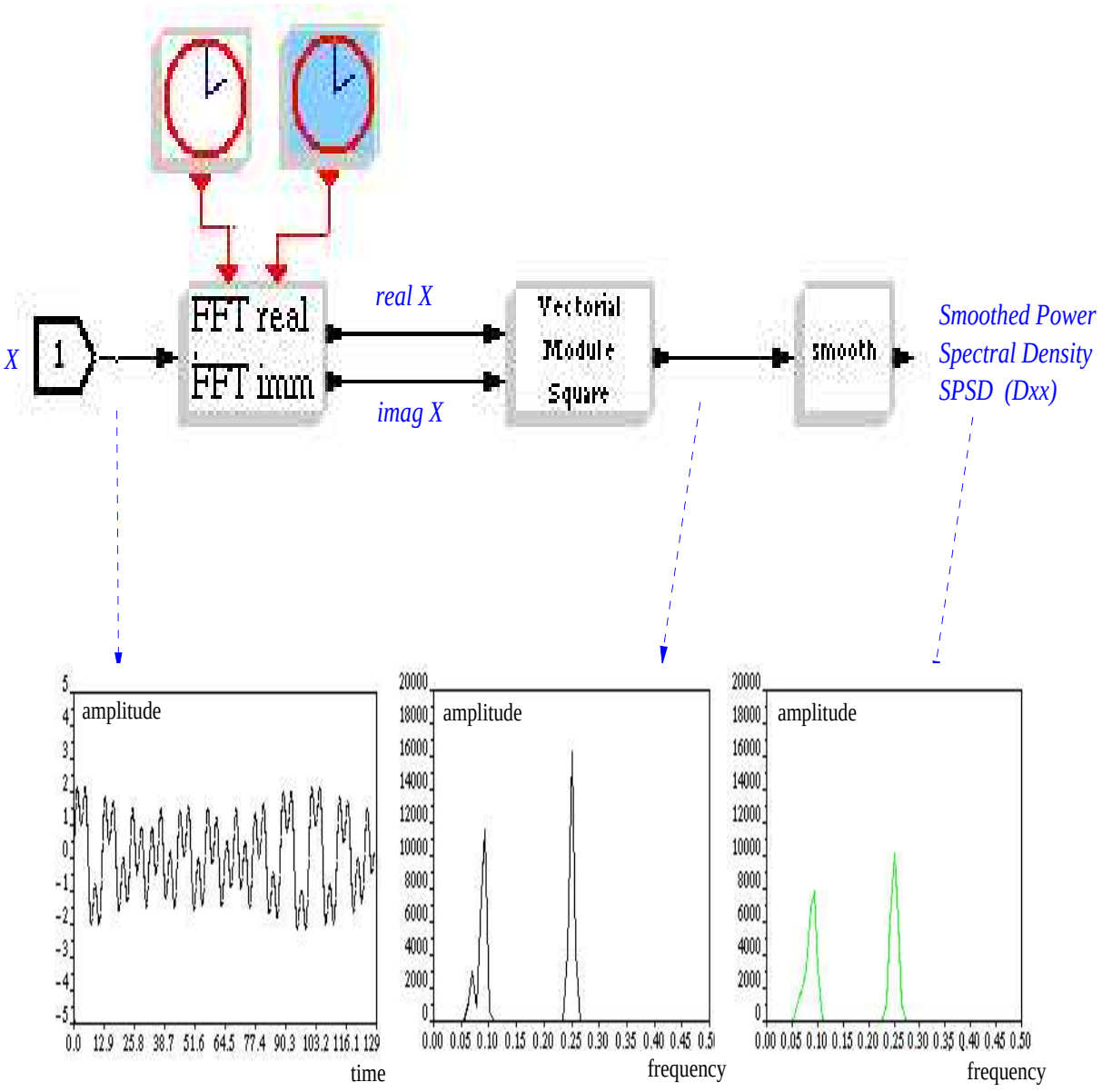


Figure 29: *Smoothing of a spectrum*

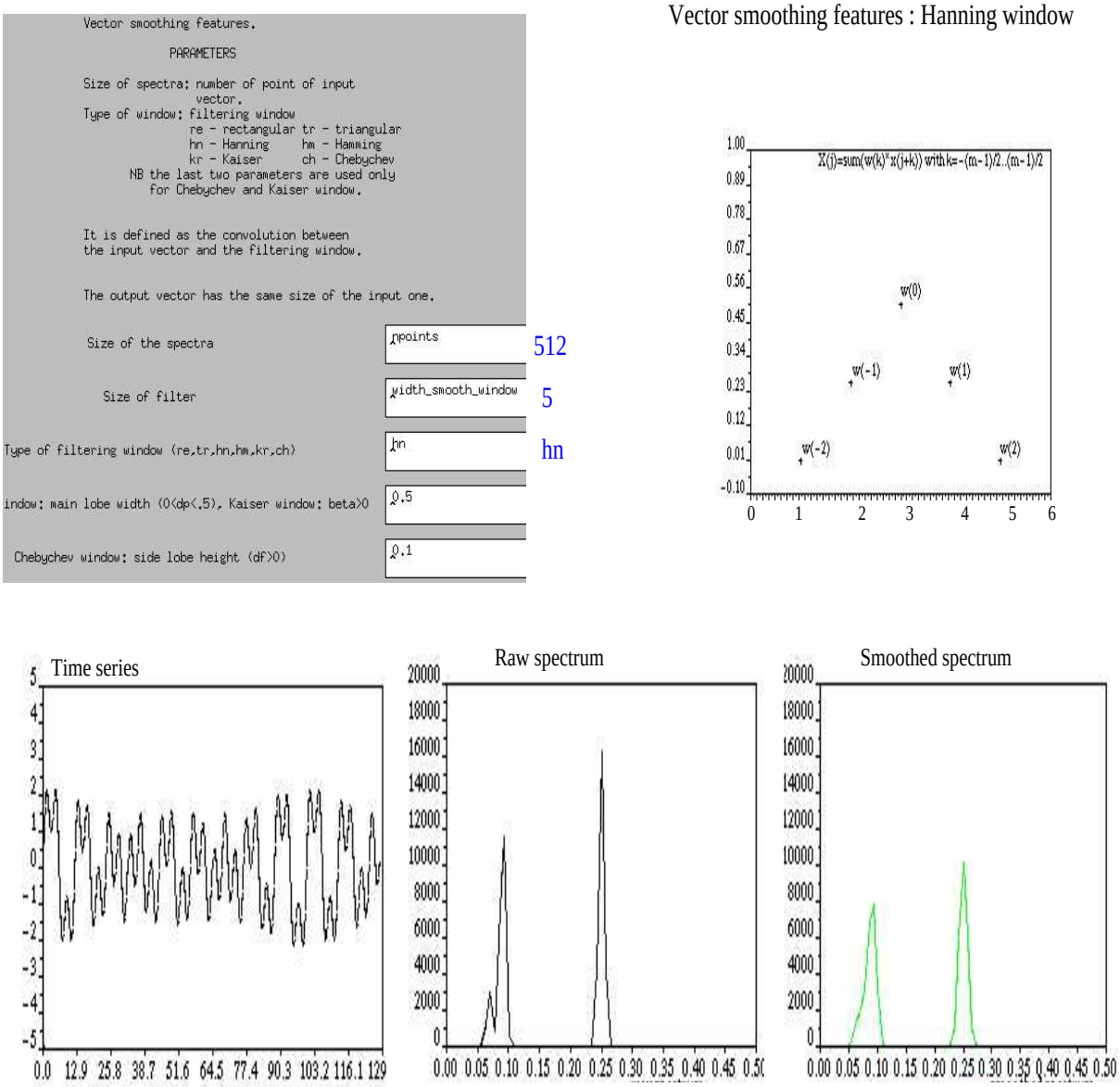


Figure 30: *Smoothing of a spectrum: Hanning window*

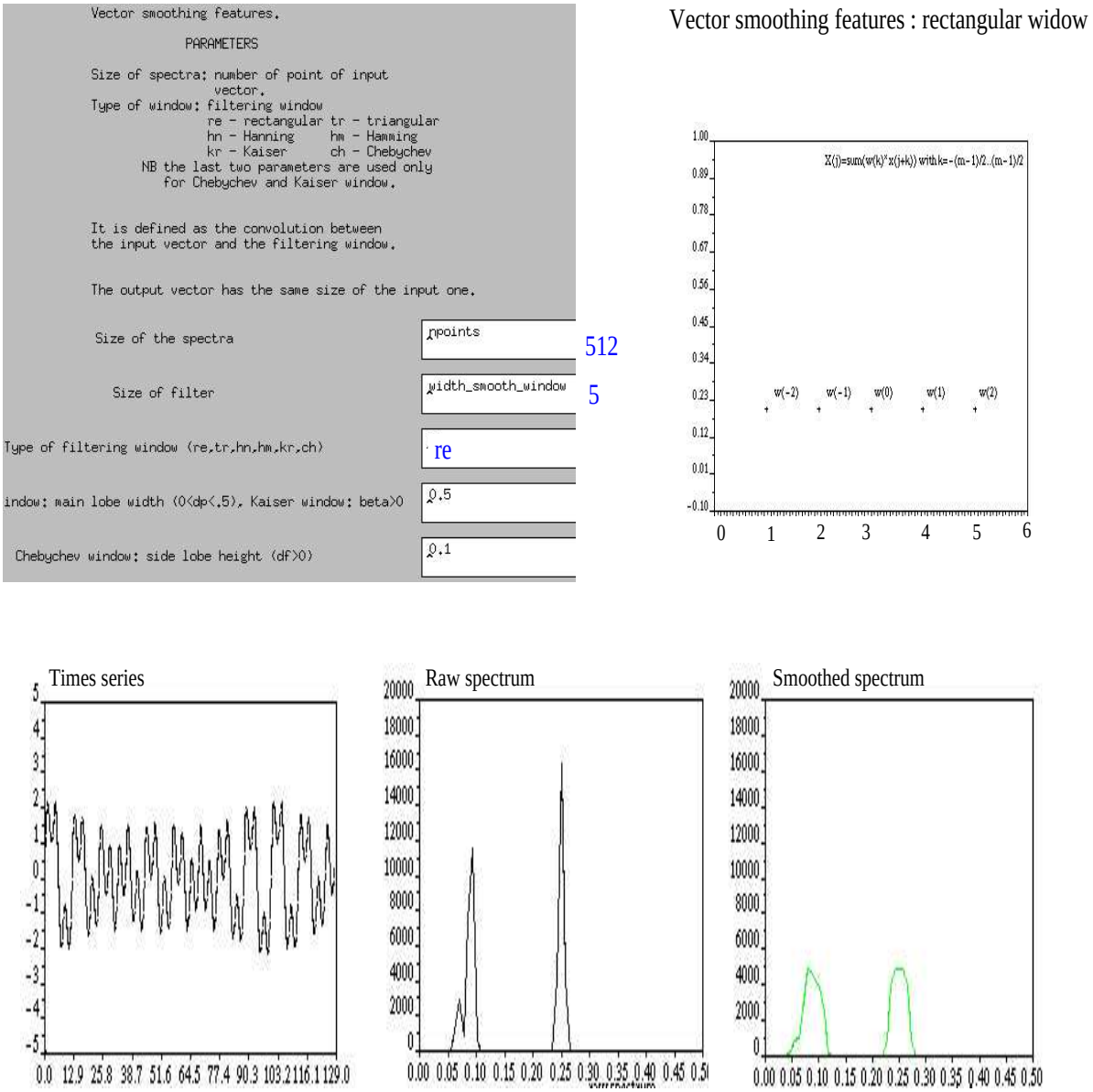


Figure 31: *Smoothing of a spectrum: rectangular window*

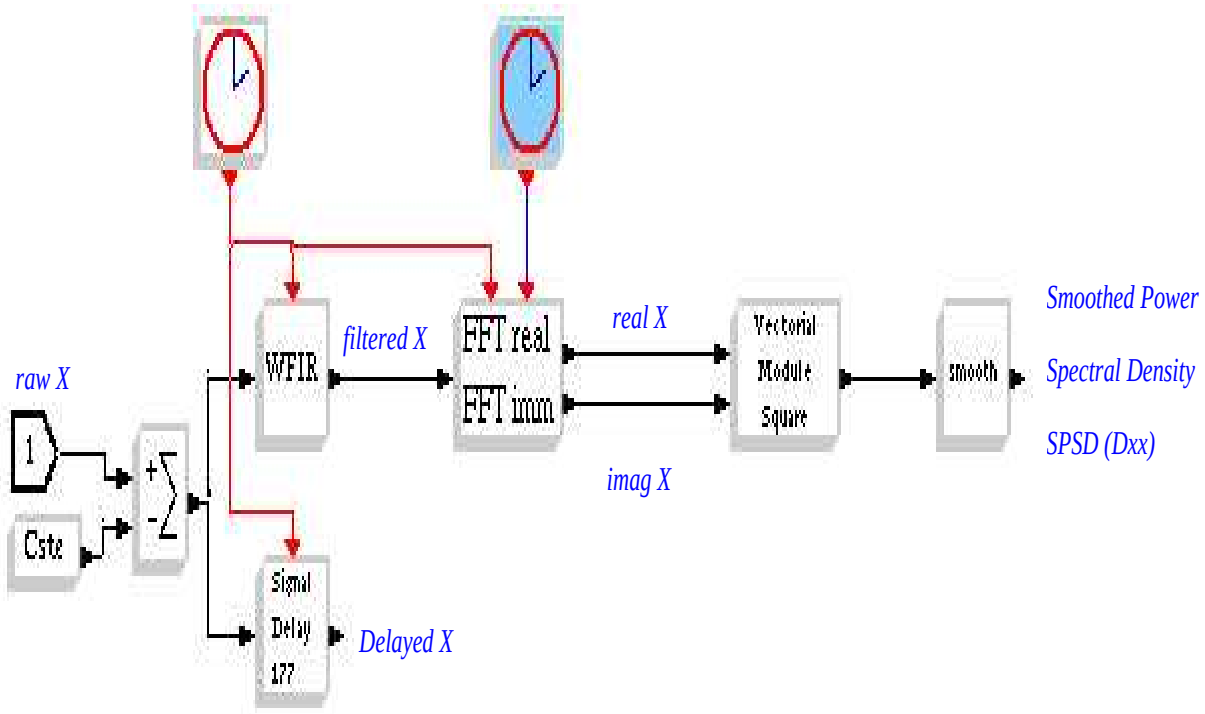


Figure 32: *Flow chart of the SPSD assessment: SCICOS diagram*

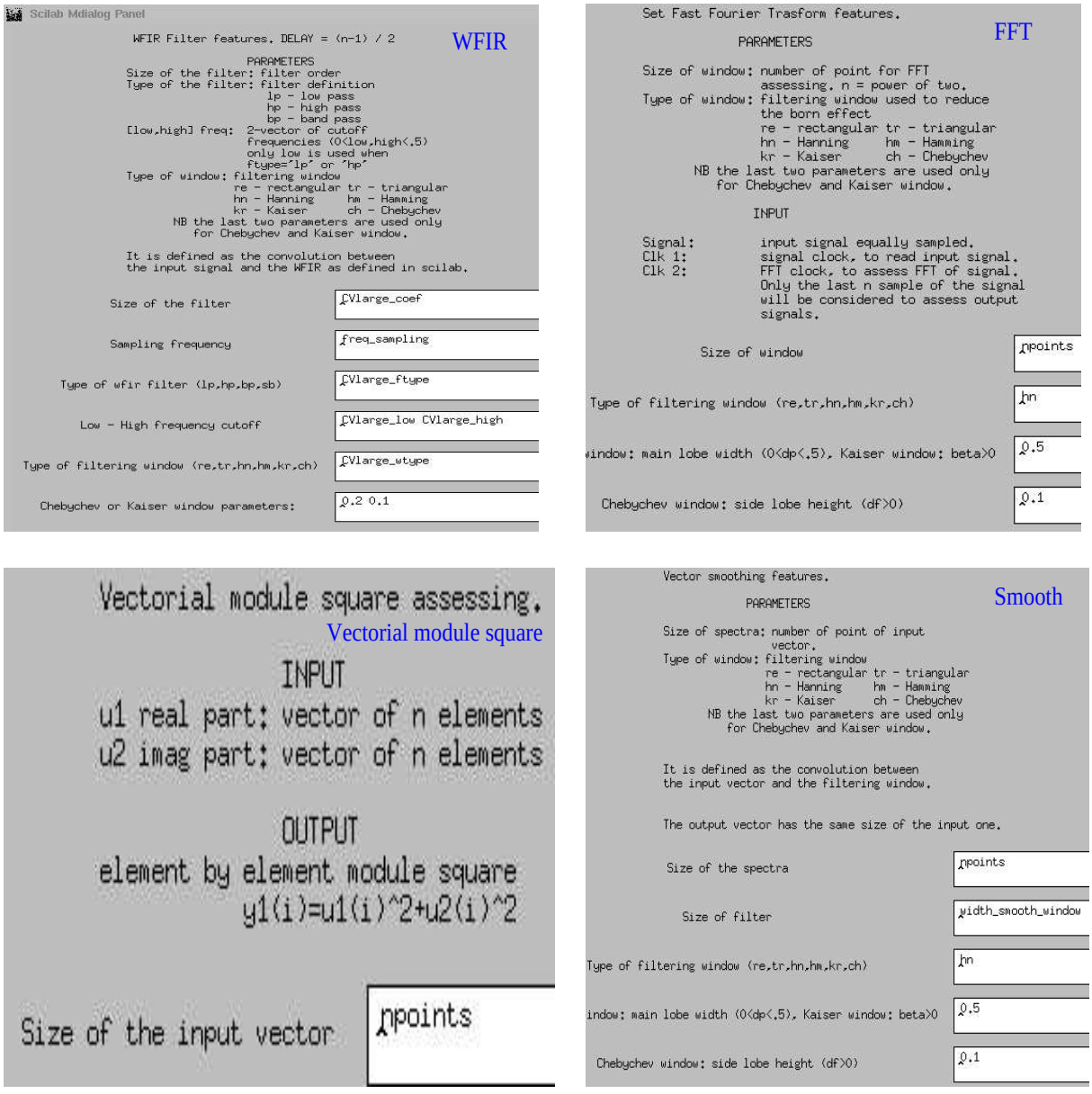


Figure 33: *Flow chart of the SPSD assessment: blocks parameters*

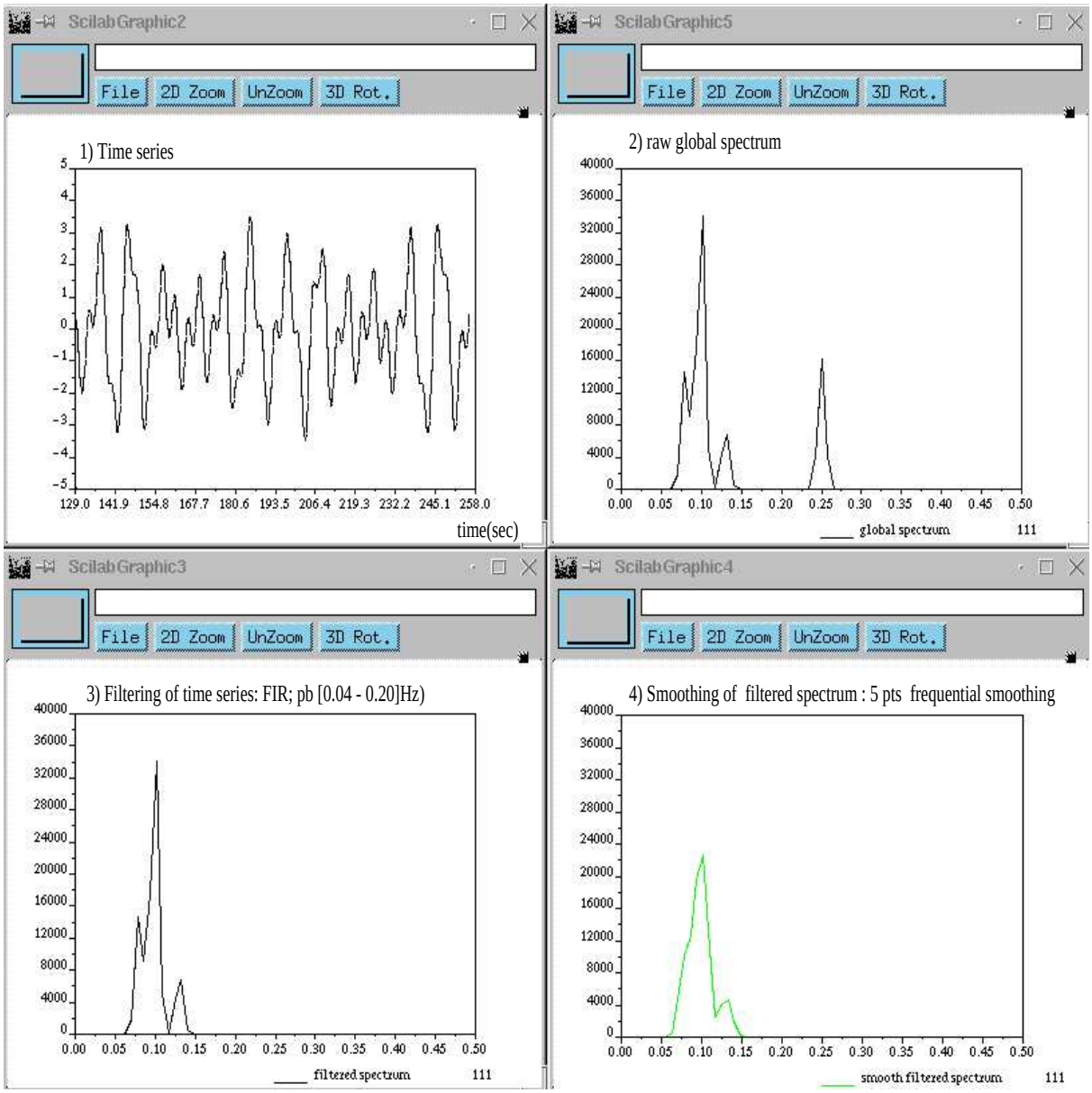


Figure 34: *From times series to Smoothed Power Spectral Density*

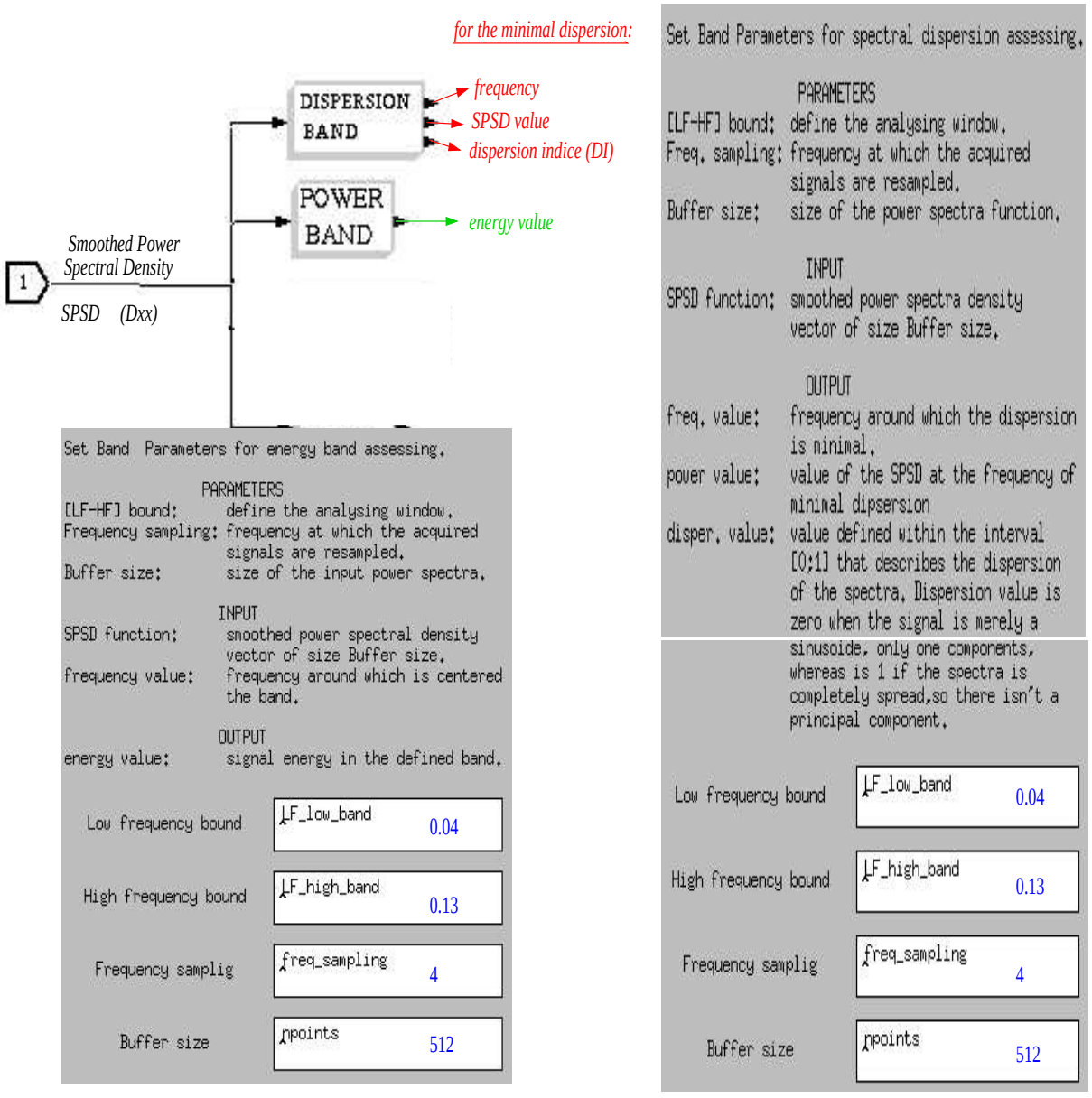


Figure 35: *Assessment of the spectral component in a Low Frequency (LF) fixed band*

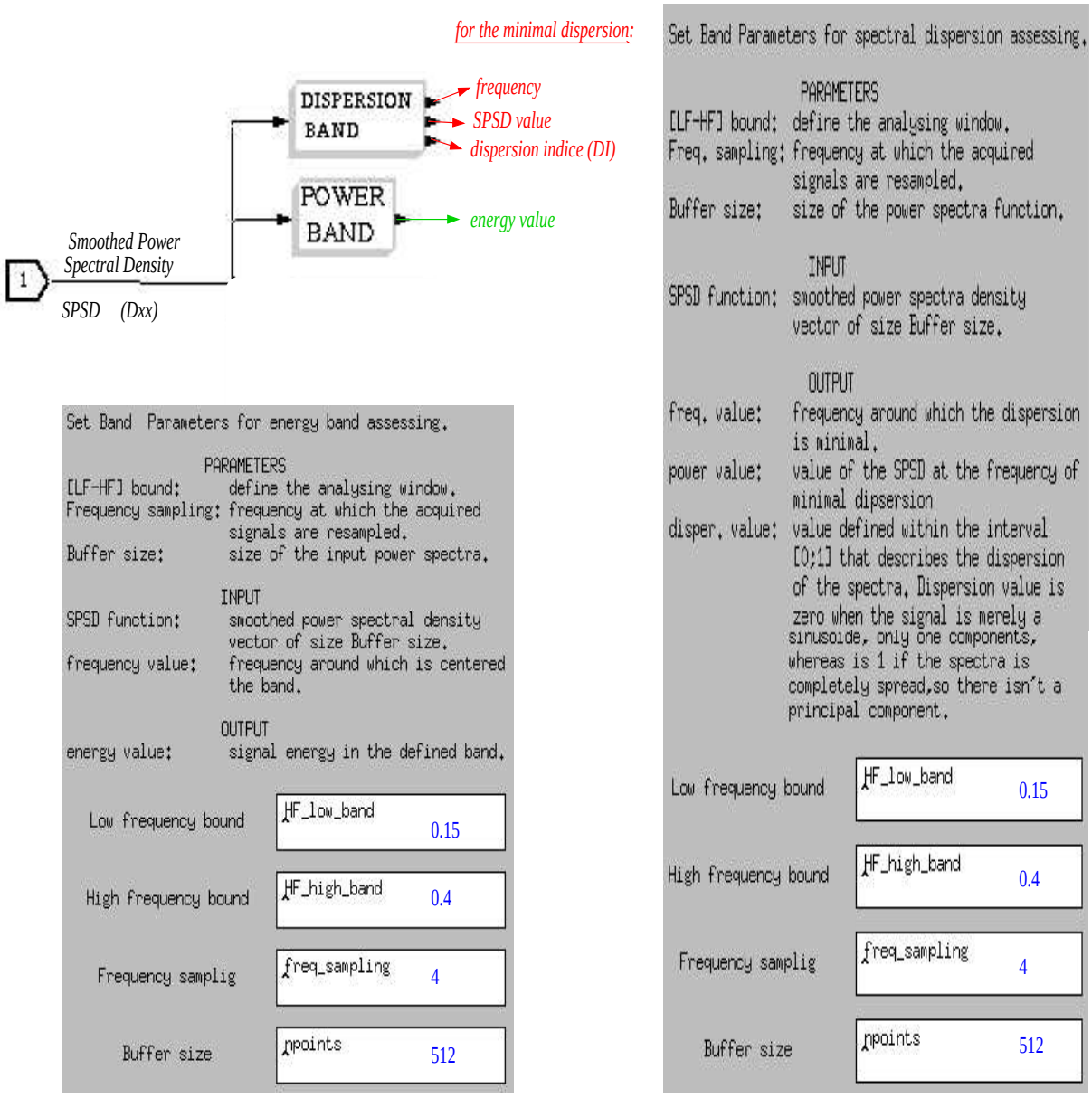


Figure 36: *Assessment of the spectral component in a High Frequency (HF) fixed band*

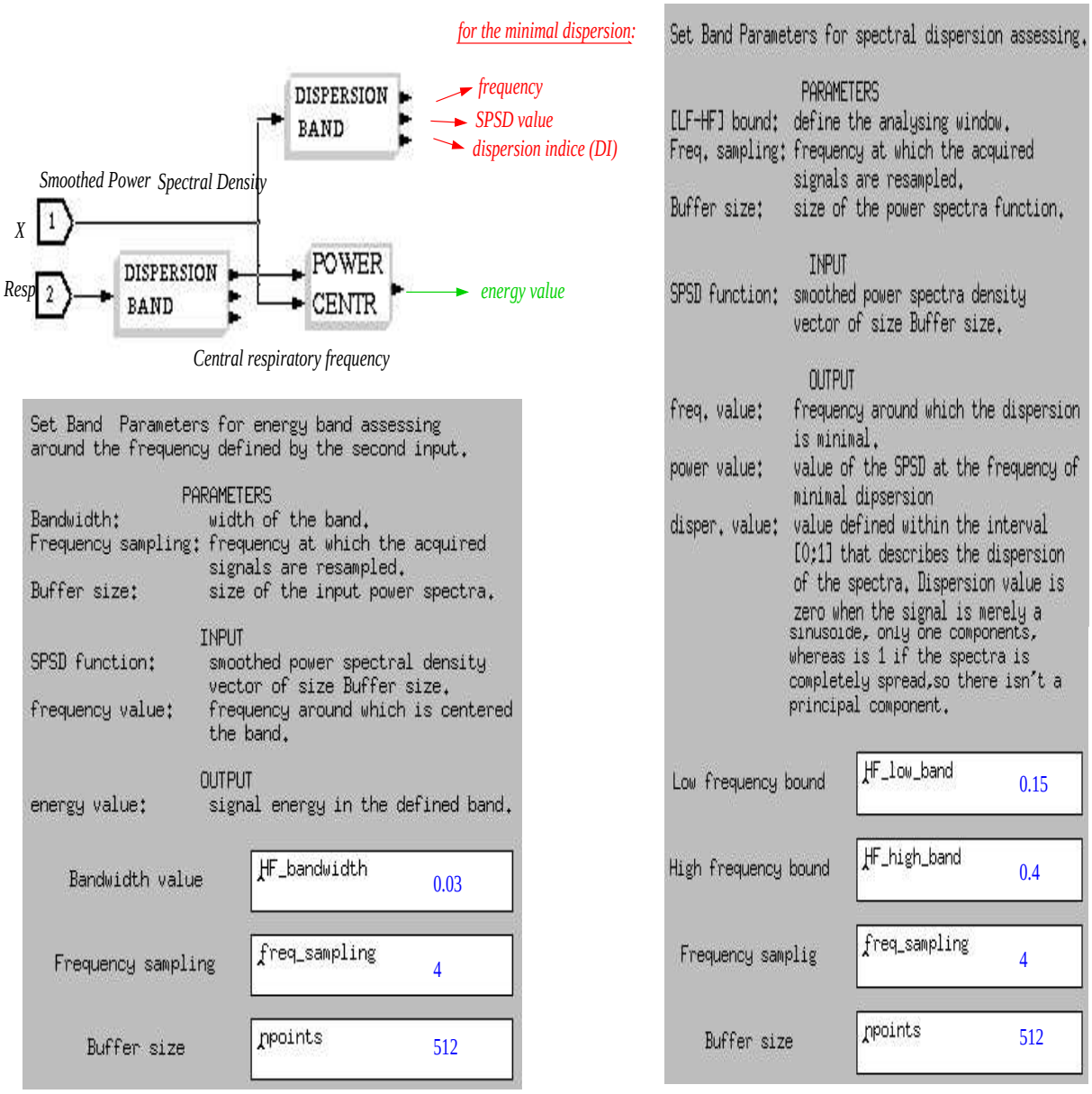


Figure 37: *Assessment of the spectral component in a bandwidth centered on the central respiratory frequency*

4.3 Spectral Baroreflex Sensitivity: Coherence and Gain assessment

4.3.1 Spectral Coherence and Gain assessment on the whole spectrum

The figures 38 and 39 explain the flow chart of the Gain analyse. The program computes the *Cross Spectral Density* between the two X and Y signals, then, after smoothing the resulting spectrum, it delivers the *Coherence function* and the *Transfer function* between their two SPSD. On the figure 40, the Gain between X and Y can be taken into account in the frequency band [0.05 - 0.15]Hz, because the corresponding Coherence is equal to 1. It is usually accepted that a coherence greater than 0.5 assumes the hypothesis of a certain linearity between two spectra, and the values between 0.5 and 1 measure the degree of linear correlation between them. Indeed, the Gain evaluation assumes the hypothesis of linearity.

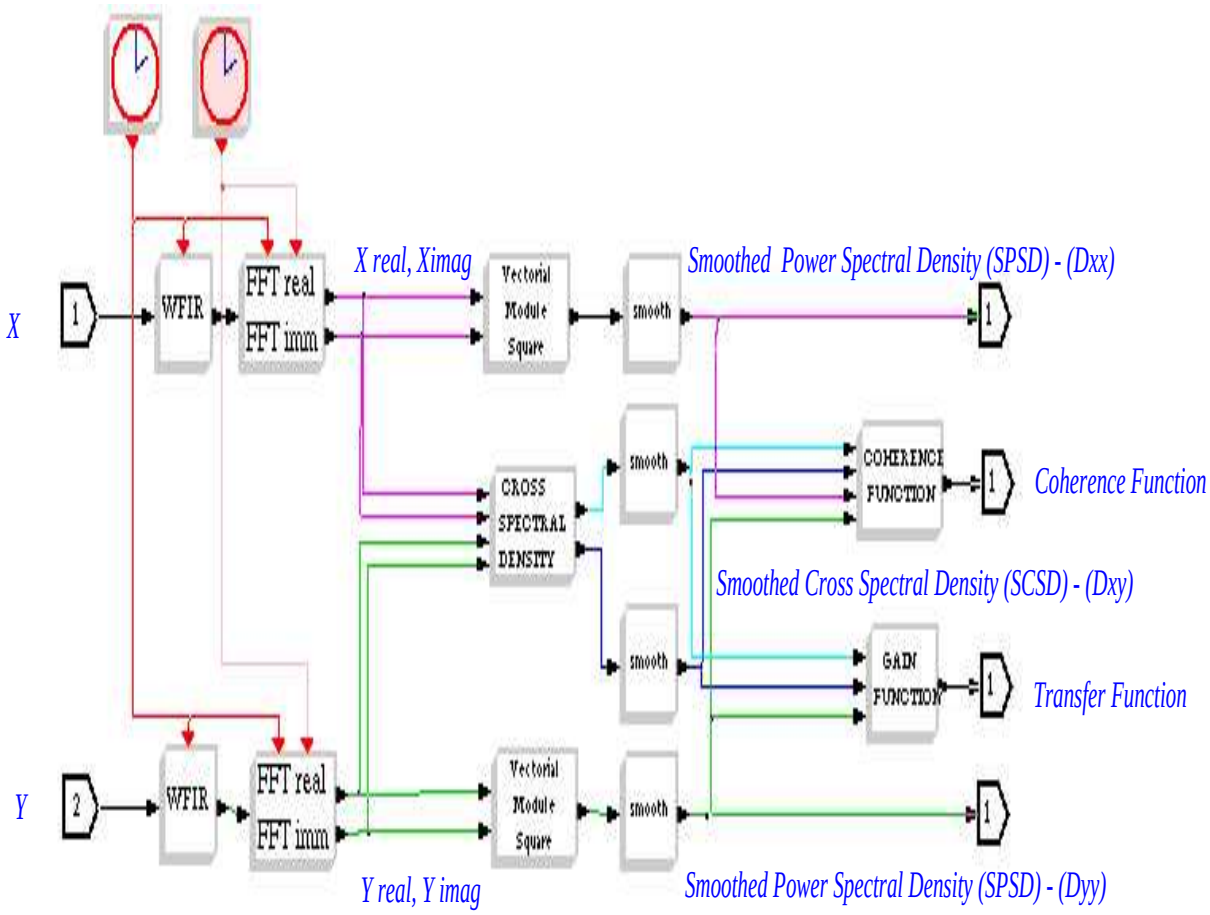


Figure 38: *Flow chart of the Gain assessment: SCICOS blocks*

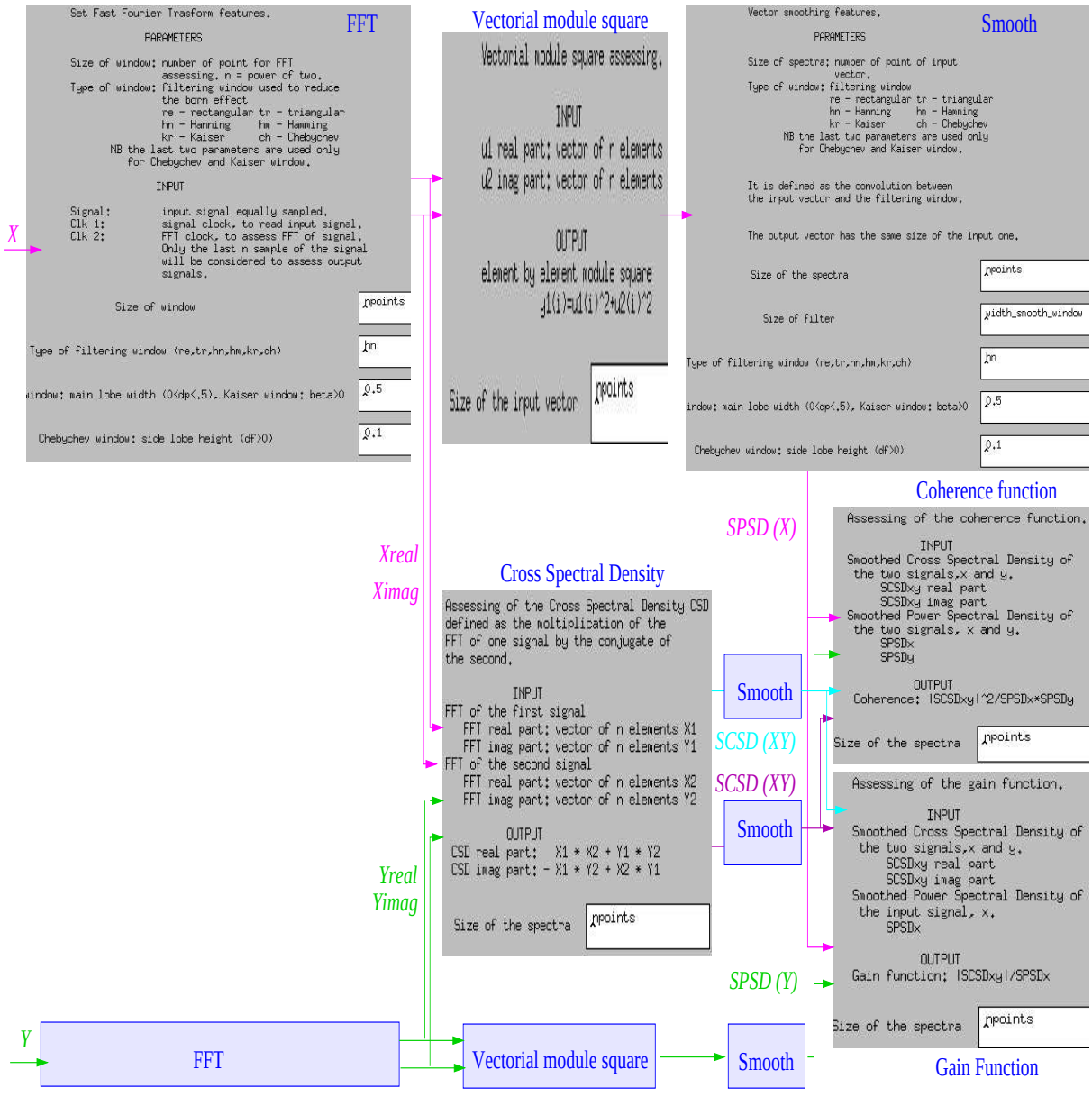


Figure 39: *Flow chart of the Gain assessment: dialog panels of the blocks*

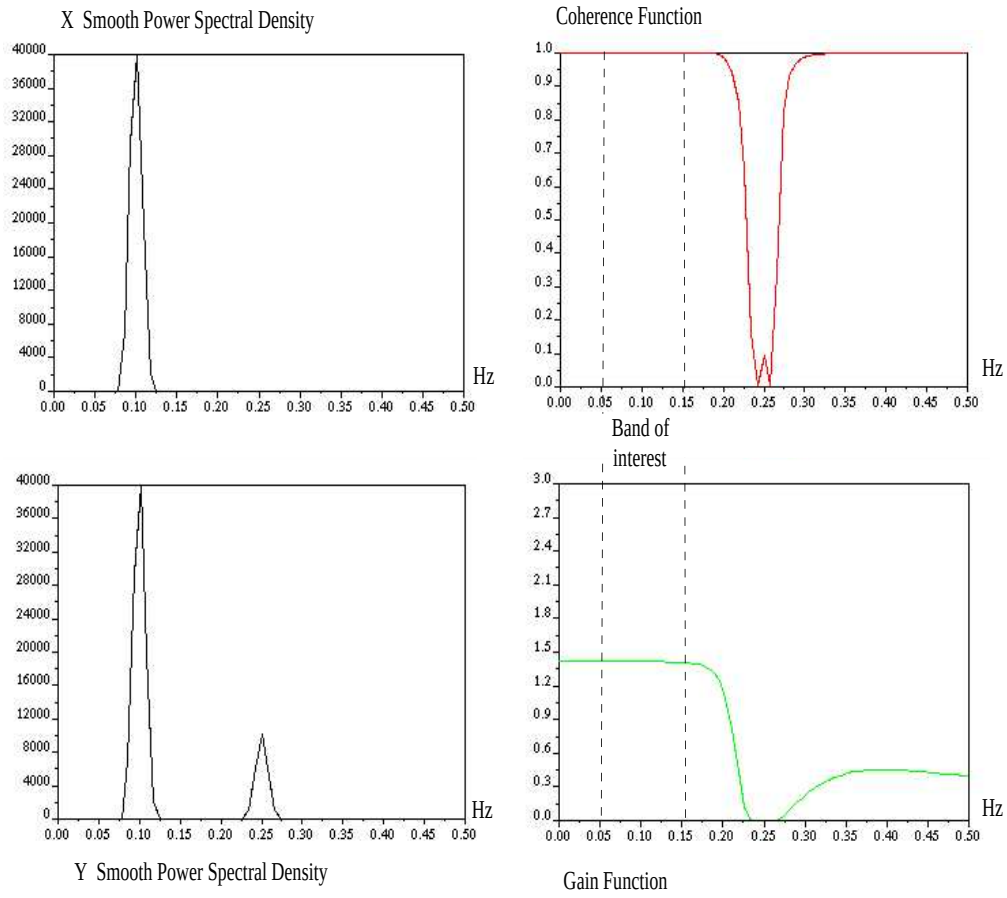
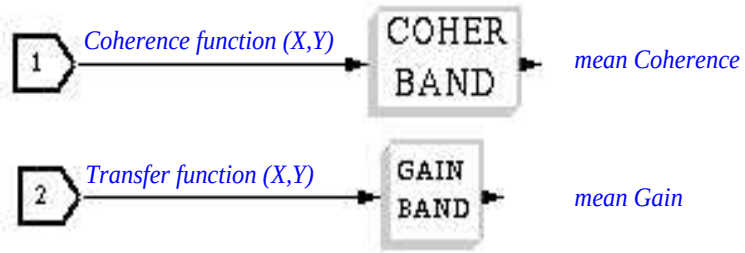


Figure 40: *Spectral features of the SPSD, Coherence and gain*

4.3.2 Spectral Coherence and Gain assessment in a frequency band

The same methodology applied to the SPSD averaging in a frequency band is replicated for the Coherence and Gain averaging. The *Coher Band* and *Gain Band* blocks used for a fixed band (Fig 41) are replaced by the *Coher Centr* and the*Gain centr* blocks when using the actual respiratory signal (Fig 42)



COHER BAND

```
Set Band  Parameters for coherence band assessing.

                PARAMETERS
[LF-HF] bound:      define the analysing window.
Minimal coherence:  minimal coherence value above which
                    the linear hypothesis is rejected.
Frequency sampling: frequency at which the acquired
                    signals are resampled.
Input size:         size of the input coherence function.

                INPUT
coherence function: vector of size Buffer size.
frequency value:    frequency around which is centered
                    the band.

                OUTPUT
coherence value:    mean coherence value in the defined
                    band. If at least one value, in the
                    defined band, is litter than Minimal
                    coherence value the mean is annulled.
```

Low frequency bound	LF_low_band_bound	0.04
High frequency bound	LF_high_band_bound	0.13
Minimal coherence value	LF_minimal_coherence	0.5
Frequency samplig	freq_sampling	4
Buffer size	npoints	512

GAIN BAND

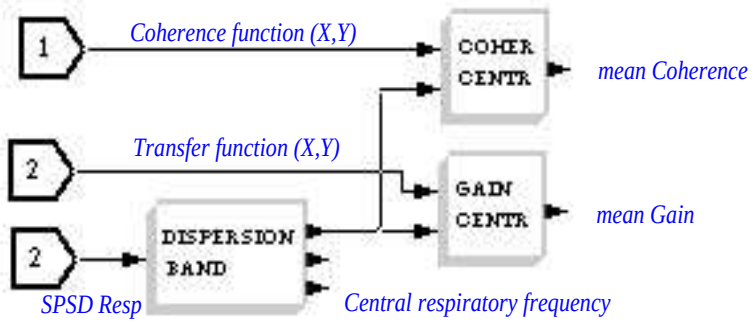
```
Set Band  Parameters for  averaged gain assessing.

                PARAMETERS
[LF-HF] bound:      define the analysing window.
Frequency sampling: frequency at which the acquired
                    signals are resampled.
Input size:         size of the input gain function.

                INPUT
Gain function:      vector of size Buffer size.

                OUTPUT
Mean gain:          mean value of the gain function
                    in the band.
```

Low Frequency bound	LF_low_band_bound	0.04
High Frequency bound	LF_high_band_bound	0.13
Frequency samplig	freq_sampling	4
Buffer size	npoints	512

Figure 41: *Mean Coherence and Gain assessment in the LF fixed band*

4.3.3 The whole spectral Gain application

The last figure concerning the spectral analysis represents the whole application. It contains all the variables precedently described: the SPSD, Dispersion, Coherence and Gain values. And this, for all the RR-interval, SBP, DBP respiratory time series, matched two by two.



COHER CENTR

Set Band Parameters for coherence band assessing around the frequency defined by the second input.

PARAMETERS
Bandwidth: width of the band.
Minimal coherence: minimal coherence value above which the linear hypothesis is rejected.
Frequency sampling: frequency at which the acquired signals are resampled.
Buffer size: size of the input coherence function.

INPUT
coherence function: vector of size Buffer size.
frequency value: frequency around which is centered the band.

OUTPUT
coherence value: mean coherence value in the defined band. If at least one value, in the defined band, is litter than Minimal coherence value the mean is annulled.

Bandwidth value	HF_bandwidth	0.03
Minimal coherence value	HF_minimal_coherence	0.5
Frequency samplig	freq_sampling	4
Buffer size	npoints	512

GAIN CENTR

Set Band Parameters for averaged gain assessing in the band centered around the frequency defined by the second input.

PARAMETERS
Bandwidth: width of the band.
Frequency sampling: frequency at which the acquired signals are resampled
Buffer size: size of the input gain function.

INPUT
Gain function: vector of size Buffer size.
Frequency value: frequency around which is centered the band.

OUTPUT
Mean gain: mean value of the gain function in the band.

Bandwidth value	HF_bandwidth	0.5
Signal Frequency sampling	freq_sampling	4
Spectral Buffer size	npoints	512

Figure 42: *Mean Coherence and Gain assessment in a bandwidth centered on the central respiratory frequency*

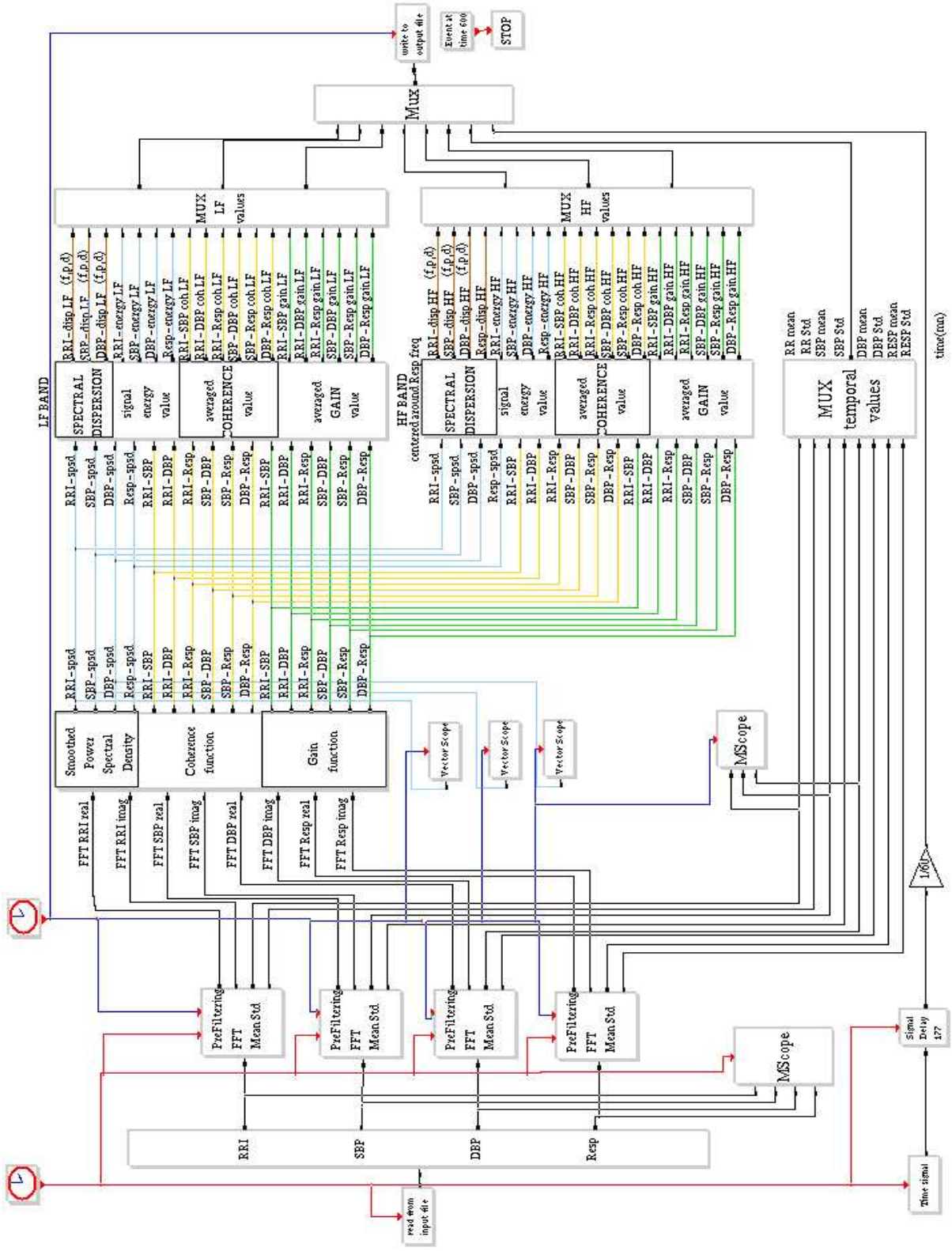


Figure 43: *The whole spectral application to cardiovascular and respiratory time series*

5 Time-varying analysis of cardiovascular and respiratory time series

5.1 Presentation

The Complex DeModulation (CDM) and the Smoothed Pseudo Wigner-Ville distribution (SPWVD) are the two methods analysed here. They give an estimation of the instantaneous spectral parameters: frequency and amplitude. The CDM gives a measure of the phase relationship between the respiratory activity and the high frequency oscillation in the cardiovascular time series. The SPWVD provides useful indexes, directly related to the noise present in the signals and to the spectral dispersion, which give an estimation of the reliability of the instantaneous parameters, frequency and phase.
The first step of the CDM and the SPWVD methods is the same, using the Hilbert Transform to obtain the complex representation of the signals. We will first present this step, then we will present CDM and SPWVD separately.

5.2 Complex Transformation of the cardiovascular time series

Four filters are applied to the cardiovascular time series; the first 3 filters are used to keep only the oscillations of interest in the spectrum; the last is the Hilbert filter. The figure 44 shows this sequence.

Prefiltering First, we substract the mean of the signal, simply represented here by a constante. Secondly, we apply a large band-pass WFIR to keep only the oscillations between 0.04Hz and 0.40 Hz, corresponding to the HF and LF frequency bands.
Thirdly, we apply a more narrow band-pass WFIR, to separately focuse on the HF and on the LF band. Concerning the HF, we can imagine several breathing conditions. A spontaneous breathing generally presents a large dispersion of the spectrum and implies a large respiratory band definition, usually fixed to [0.13 - 0.4] Hz (Task Force definition). When having the respiratory signal, it is possible to know the mean breathing rate and so, to define more precisely the HF band. The paced breathing condition is the best to improve the results of the methods, because a very narrow band can be delimited. The figure 45, shows an example of paced breathing at 0.15Hz, allowing a band-pass between 0.12 and 0.18 Hz. The parameters of the two WFIR are represented on the figure 45
The LF band is usually fixed to [0.04 - 0.4] Hz, according to the Task Force. In the case of the CDM application, the demodulation is made using a sinusoid generator at 0.8Hz. The difference between HF and LF will be illustrated by the figure 51 and 53.

Hilbert Transform The Hilbert Transform gives the complex representation of the real signal X, i.e its real and imaginary parts, $realX$ and $imagX$, as shown on the figure 46. It shifts the signal by 90 degrees, so that a cosinus becomes a sinus. The real part of X is simply resynchronised with the imaginary part by the *Signal Delay* block.

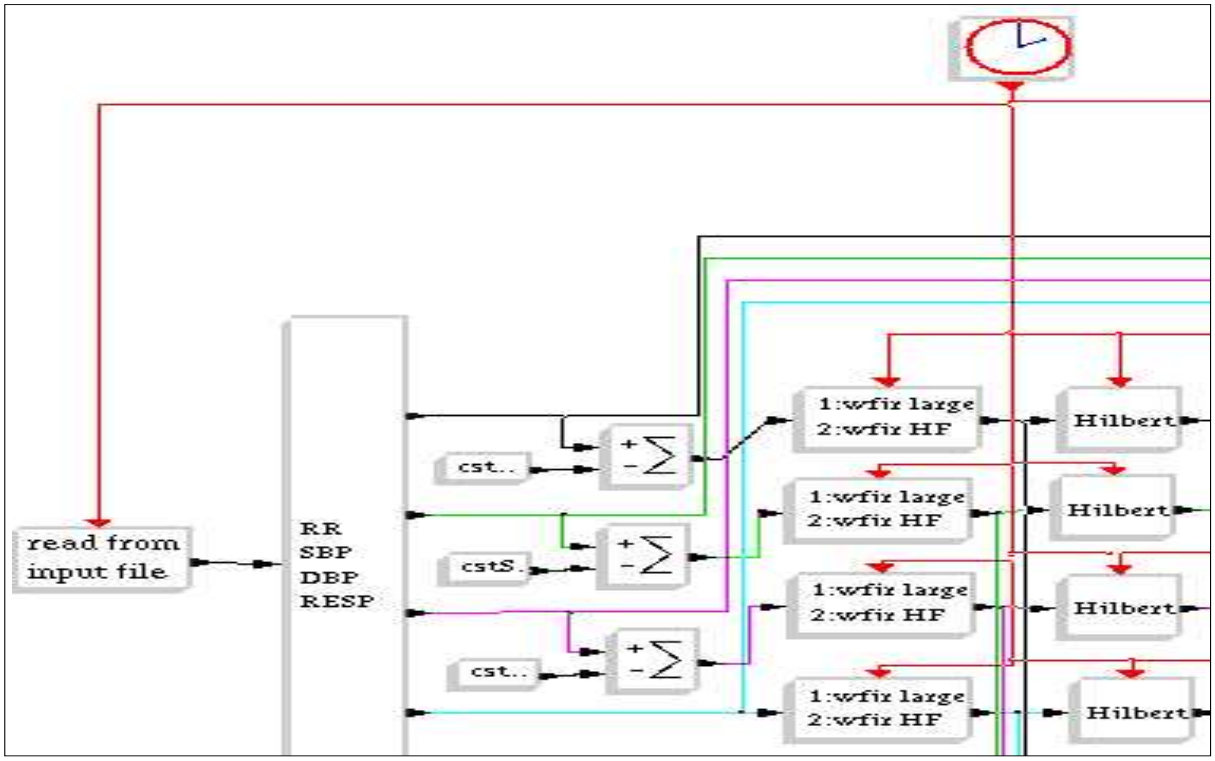


Figure 44: *Preprocessing: successive filtering providing the complex representation of the signals*

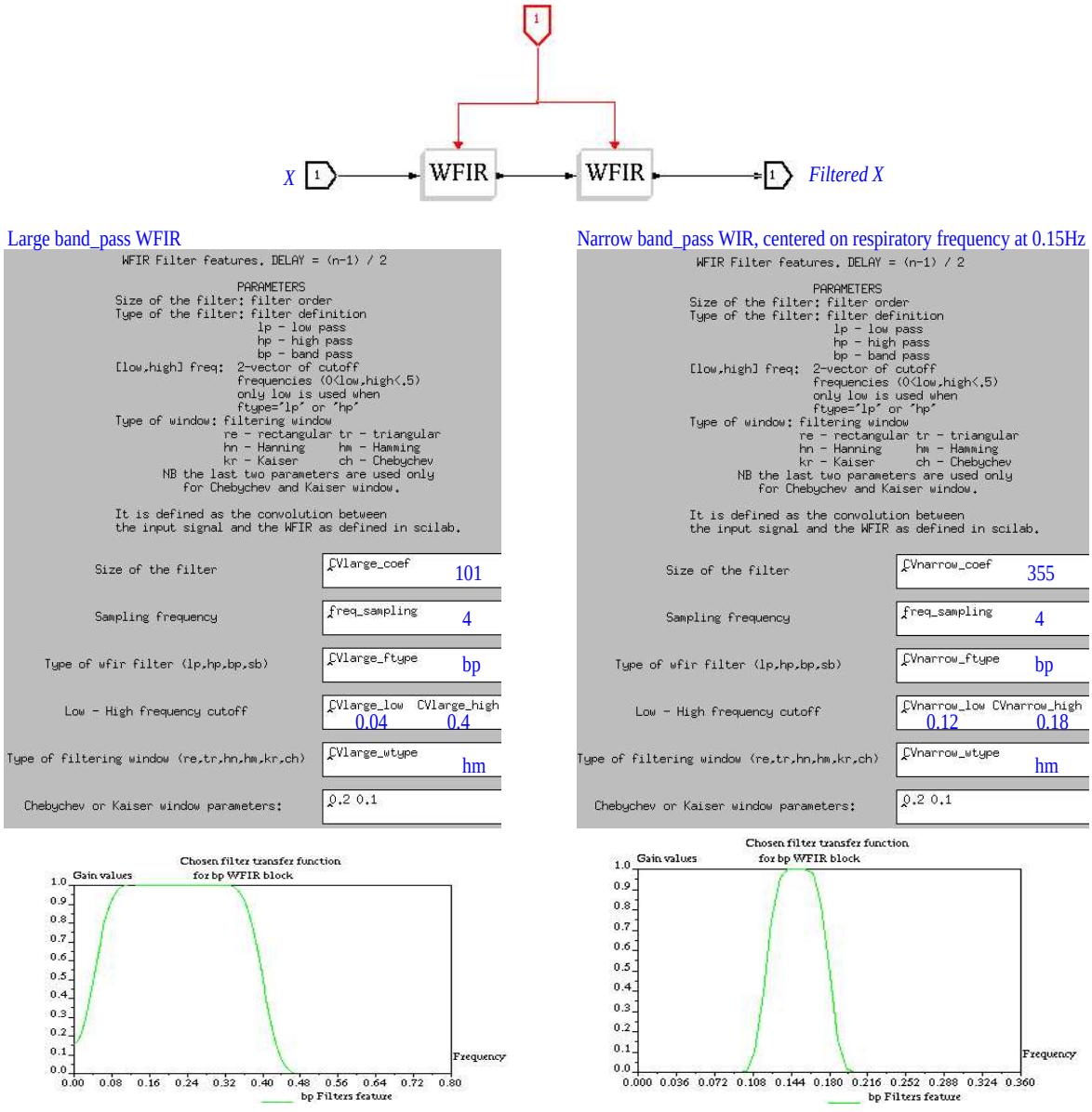
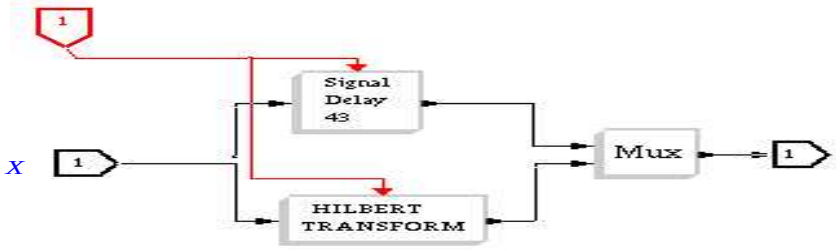


Figure 45: *Parameters of the two WFIR*



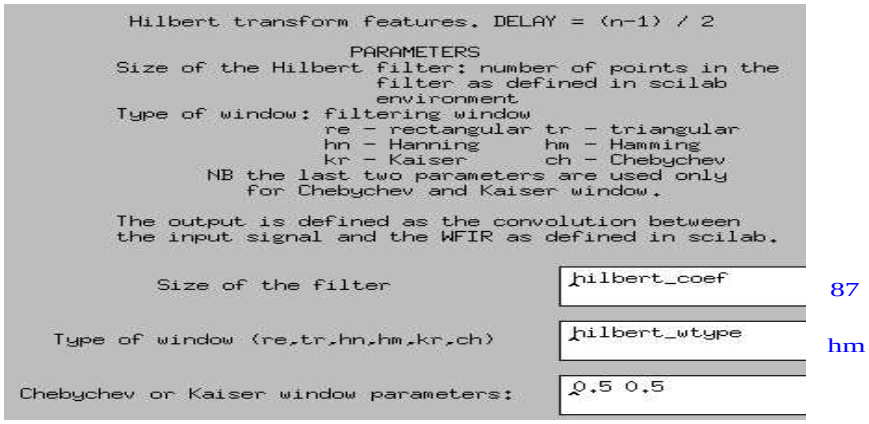


Figure 46: *Feature of the Hilbert filter*

5.3 Complex DeModulation (CDM)

5.3.1 Description of the blocks

The figure 47 shows the whole flow chart of the CDM following the Hilbert Transform. First, the *Phasor* block, described in the figure 48, is linked to the respiratory signal. The respiratory phase is extracted by computing the arc-tangent of the complex signal (Fig. 48). The *Continuous phase* block allows to avoid abrupt changes between π and $-\pi$.
The respiratory phase is then computed in the *CDM* block, with the complex X signal to realise the demodulation of X at the respiratory frequency (Fig. 49). As the DMC operation provides a leftward shift of all frequencies, a low-pass WFIR filter has to be applied to keep only the frequency of interest, in the *Filter LP* block. The *Phase Amplitude* block then provides the phase, as precedently described, and the amplitude, as the scalar module of the demodulation of X at the respiratory frequency.
The instantaneous frequency is obtained by the first order differentiation of the instantaneous phase (Fig. 50).

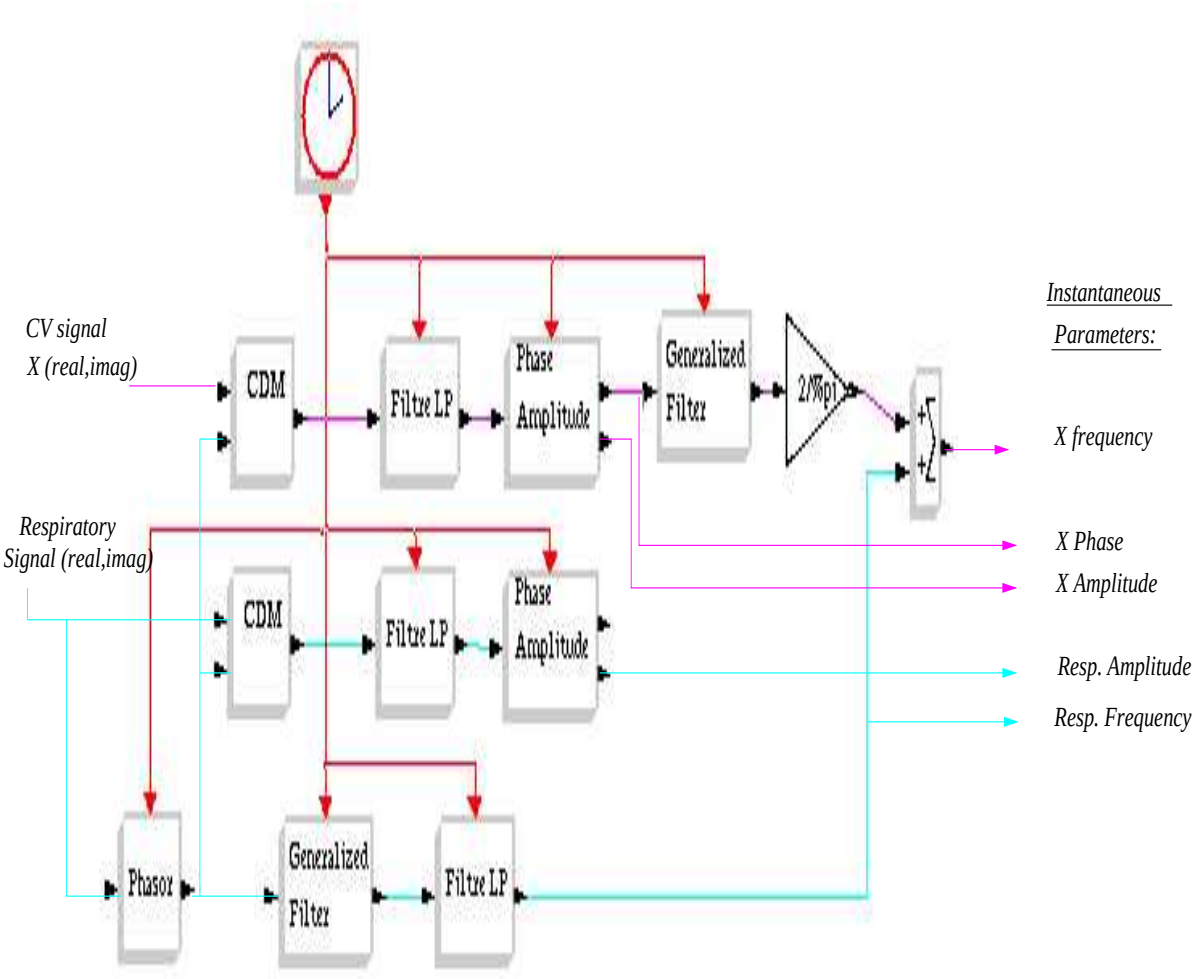
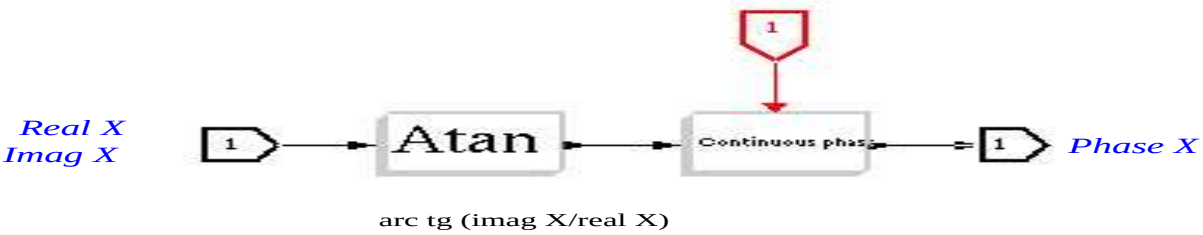


Figure 47: *CDM between the respiratory signal and a cardiovascular time series*



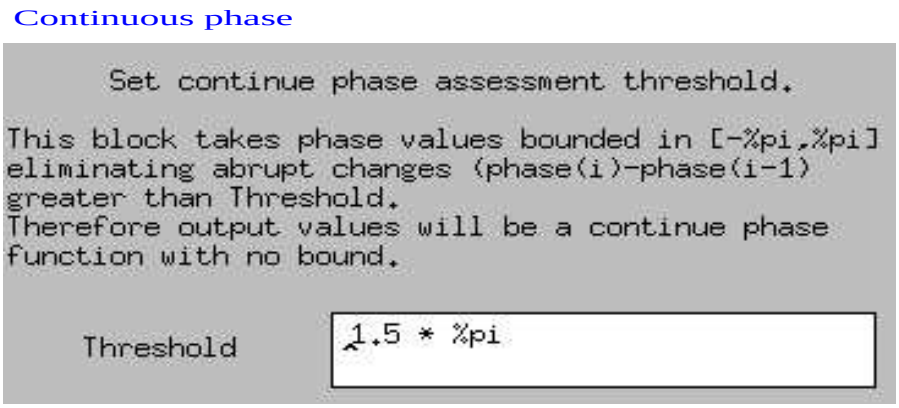


Figure 48: *Phase assessment of the respiratory signal*

5.3.2 The whole CDM application

The last figures 51, 52 and 53 show the whole applications with the three cardivascular time series, RR, SBP and DBP. The figures 51 and 52 refer to the demodulation between the cardiovascular signals and the respiratory signal fixed at 0.25 Hz. One can see on the context that the narrow WFIR has a bandwidth of [0.22 - 0.28] Hz, very tight around the central respiratory frequency (Fig. 52). The figure 53 is dedicated to the demodulation applied in the LF frequency band. In this case, the central frequency used for demodulating is provided by a sinusoid generator fixed at 0.08 Hz and we analyse the instantaneous amplitude and frequency, with some precautions related in the Research Report.

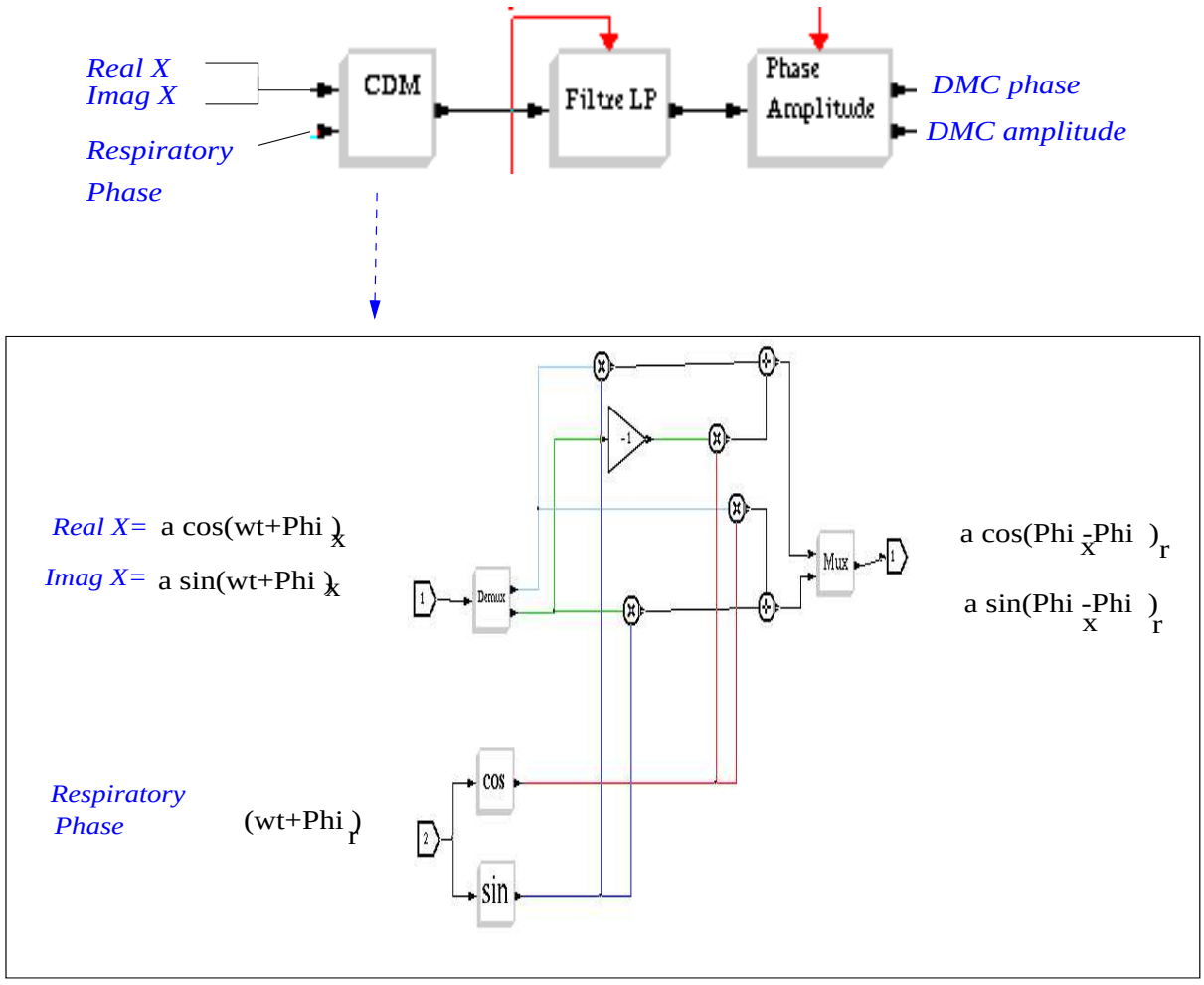


Figure 49: *Complex DeModulation (CDM)*

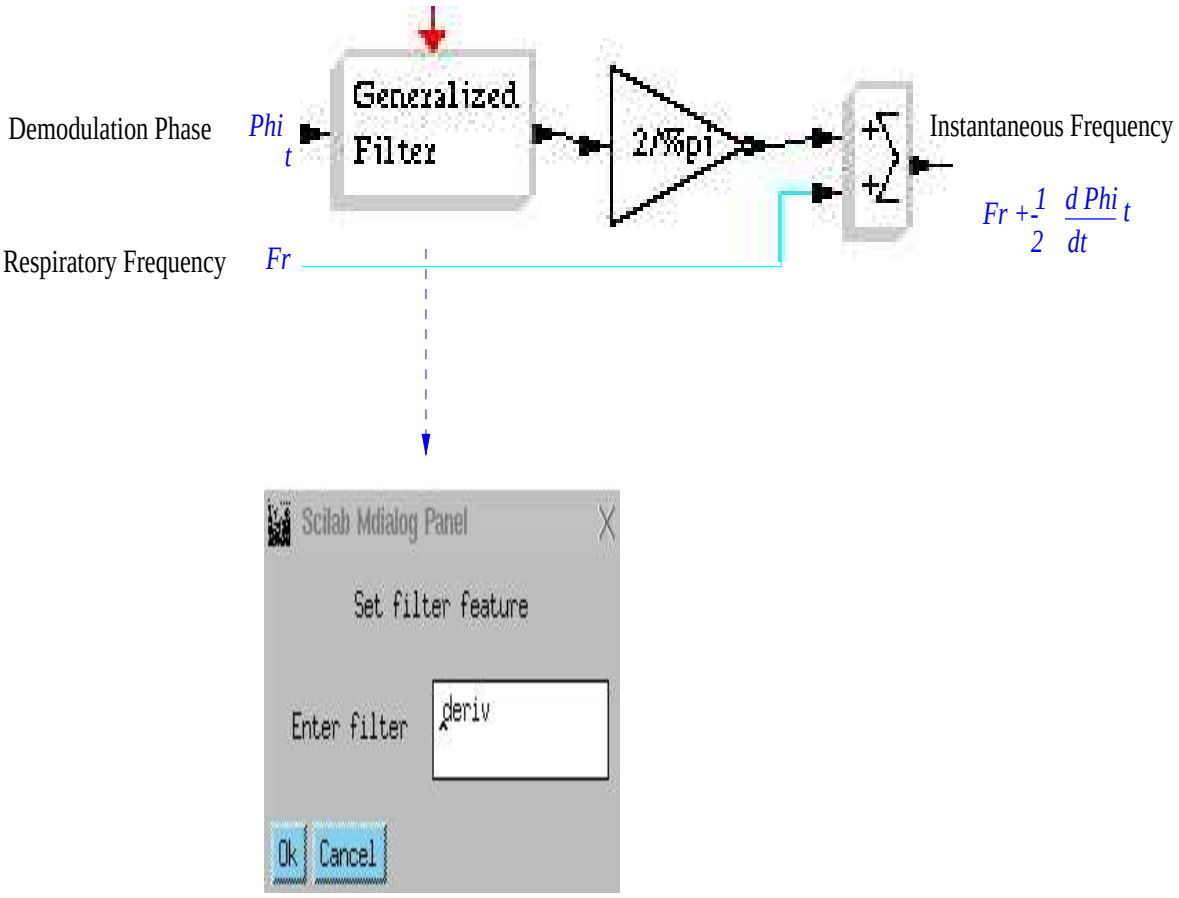


Figure 50: *From the instantaneous phase to the instantaneous frequency: first order differentiation*

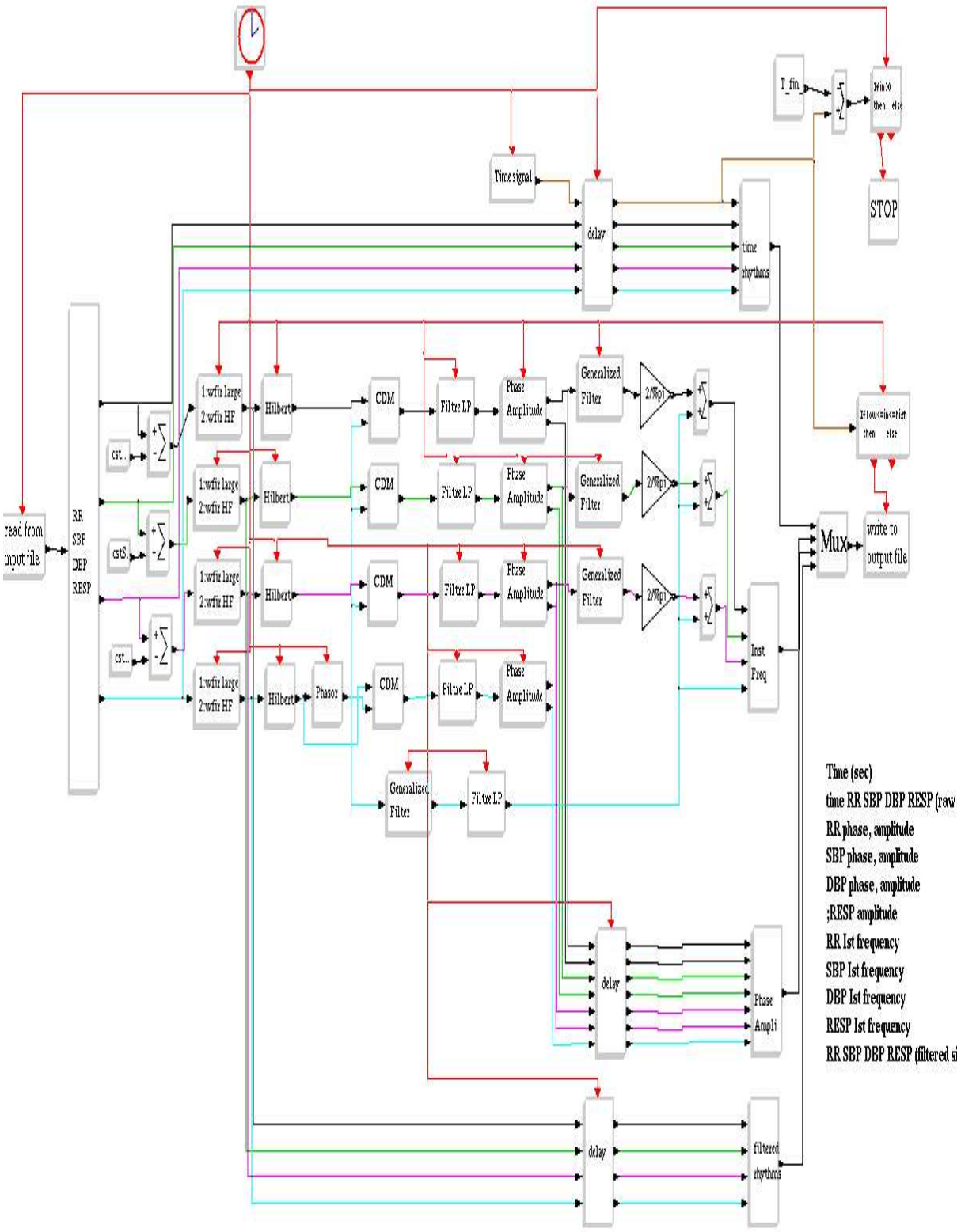


Figure 51: *The whole CDM applied at a respiratory frequency of 0.25 Hz*

```
freq_sampling=4;
cstRR=900;cstSBP=110;cstDBP=60;

WFIR large
CVlarge_coef=101;
CVlarge_delay=(CVlarge_coef-1)/2;
CVlarge_ftype='bp';
CVlarge_wtype='hm';
CVlarge_low=0.04;
CVlarge_high=0.4;

WFIR narrow: HF
CVnarrow_coef=355;
CVnarrow_delay=(CVnarrow_coef-1)/2;
CVnarrow_ftype='bp';
CVnarrow_wtype='hm';
CVnarrow_low=0.22;
CVnarrow_high=0.28;

Hilbert filter
hilbert_coef=87;
hilbert_wtype='hm';
hilbert_delay=(hilbert_coef-1)/2;

LP filter
filterLP_coef=41;
filterLP_delay=(filterLP_coef-1)/2;
filterLP_ftype='lp';
filterLP_wtype='hm';
filterLP_low=0.02;
filterLP_high=1.9;
filterLP_low_norm=filterLP_low/freq_sampling;
filterLP_high_norm=filterLP_high/freq_sampling;
[filter_LP,wfm_LP,fr_LP]=wfir('lp',filterLP_coef,[filterLP_low_norm,filterLP_high_norm],'hm',[0.5,0.5]);

Generalized filter or derivative filter: first order differentiation
deriv=[-0.000235;0.0009893;-0.002725;0.0066583;-0.0157403;0.0396824;-0.1294942;1.2608785;-1.2608785;0.129
filterDER_coef=16;
filterDER_delay=(filterDER_coef)/2;

Successive delays related to the different filters
global_delay=CVlarge_delay + CVnarrow_delay + hilbert_delay + filterLP_delay + filterDER_delay;
global_delay_sec=global_delay/freq_sampling;
```

Figure 52: *The context of the whole CDM applied at a respiratory frequency of 0.25 Hz*

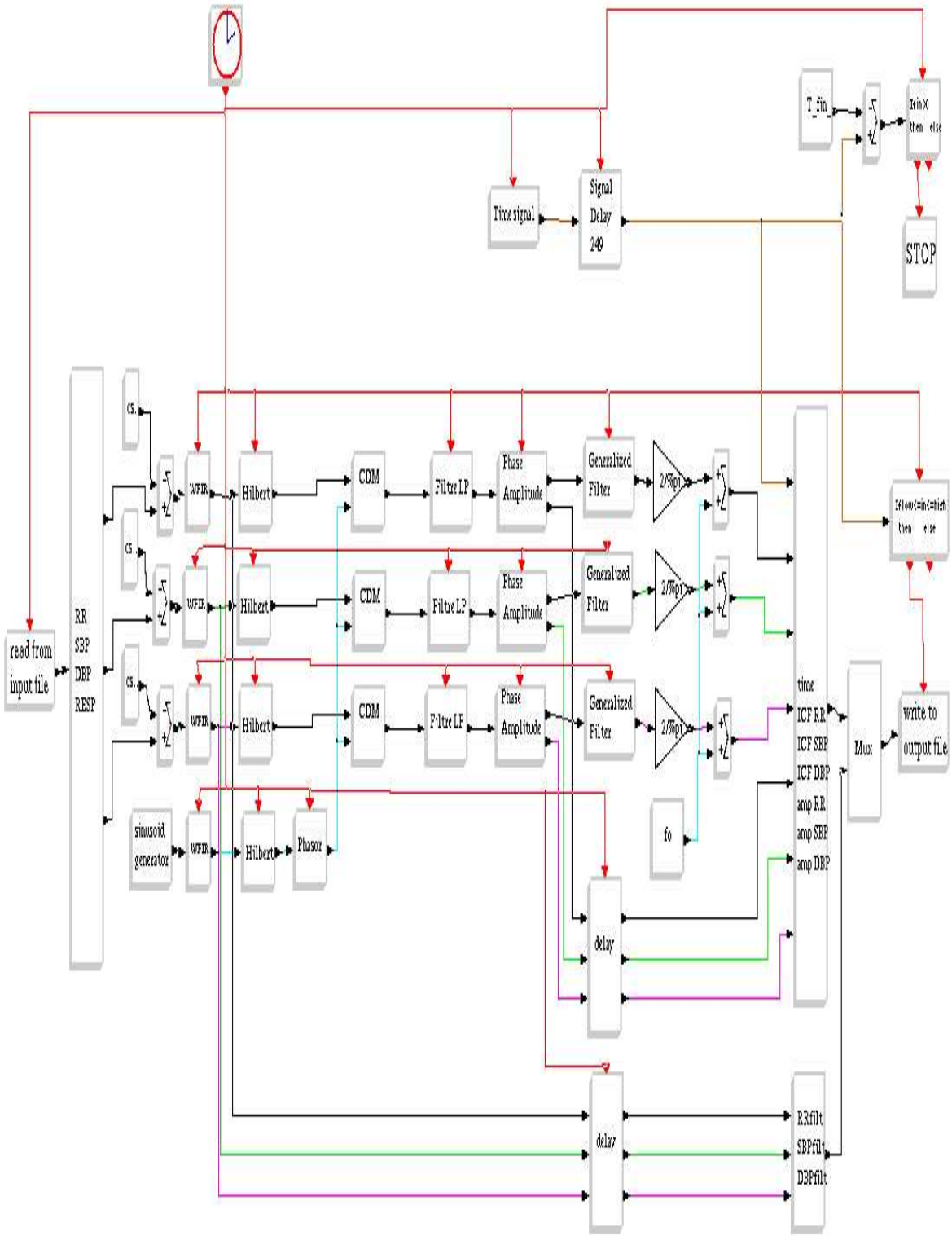


Figure 53: *The whole CDM applied in the LF band*

5.4 Smoothed Pseudo Wigner-Ville distribution (SPWVD)

5.4.1 The SCICOS implementation of the SPWVD

The figure 54 represents the general view of the SPWVD applied to the cardiovascular time series, RR, SBP and DBP and the respiratory signal. We simply notice that the raw signals and the time signal are delayed by the sum of all the filters delays. The context related to this application is described on the figure 55.

The figure 56 shows the flow chart of the SPWVD applied to the RR time series. We will decompose it into three steps: the processing before the SPWVD, the SPWVD itself, then the instantaneous parameters computation.

The preprocessing is described in the *Complex Transformation of the cardiovascular time series*. It includes the filtering of the raw signals and the Hilbert Transform which decomposes the signal into its real and imaginary parts.

The SPWVD itself corresponding to the *SPWVD* block, is represented on the figures 57 and 58. The SPWVD has time and frequency independent definitions, and is smoothed in these two directions, by the temporal and frequency parameters, found in the whole context of the figure 55, and in the figure 58. The main steps of the SPWVD are a translation of the complex signal followed by the smoothing in time and frequency, a zero padding (realized by the *vector composing* block) to improve the frequential resolution, a mirror writing and a complex FFT of the resulting matrix. The window size of the temporal smoothing is of 48 samples for a frequency sampling equal to 4 HZ and of 24 samples for a frequency sampling equal to 2 HZ.

The SPWVD Instantaneous parameters are shown on the figure 59. The *Ist Freq, Ist Disp, Ist Power* block takes the real part of the SPWVD and computes the mass center of the Wigner-Ville distribution corresponding to the instantaneous frequency, its dispersion and its integral over frequency, corresponding to the instantaneous power of the spectrum, between 0.04 - 0.04 Hz. The *IF, IDisp, IPow, IAmp filtering* block computes the square root of the instantaneous power to provide the amplitude and low-pass filters these instantaneous parameters. The *Ist Freq, Ist Disp, Ist Power BOUND* and *bounded IF, IDisp, IPow, IAmp filtering* blocks are the same computations but reduced to the HF and LF bands. So, the comparaison between the whole power and the HF or LF power allows to compute the noise estimation of the signal. As the dispersion can take negative values (a SPWVD particularity), the estimation of the dispersion is done in absolute values. The dispersion and noise estimate together give indications on the reliability of the instantaneous frequency (for SPWVD and CDM) and phase (for CDM).

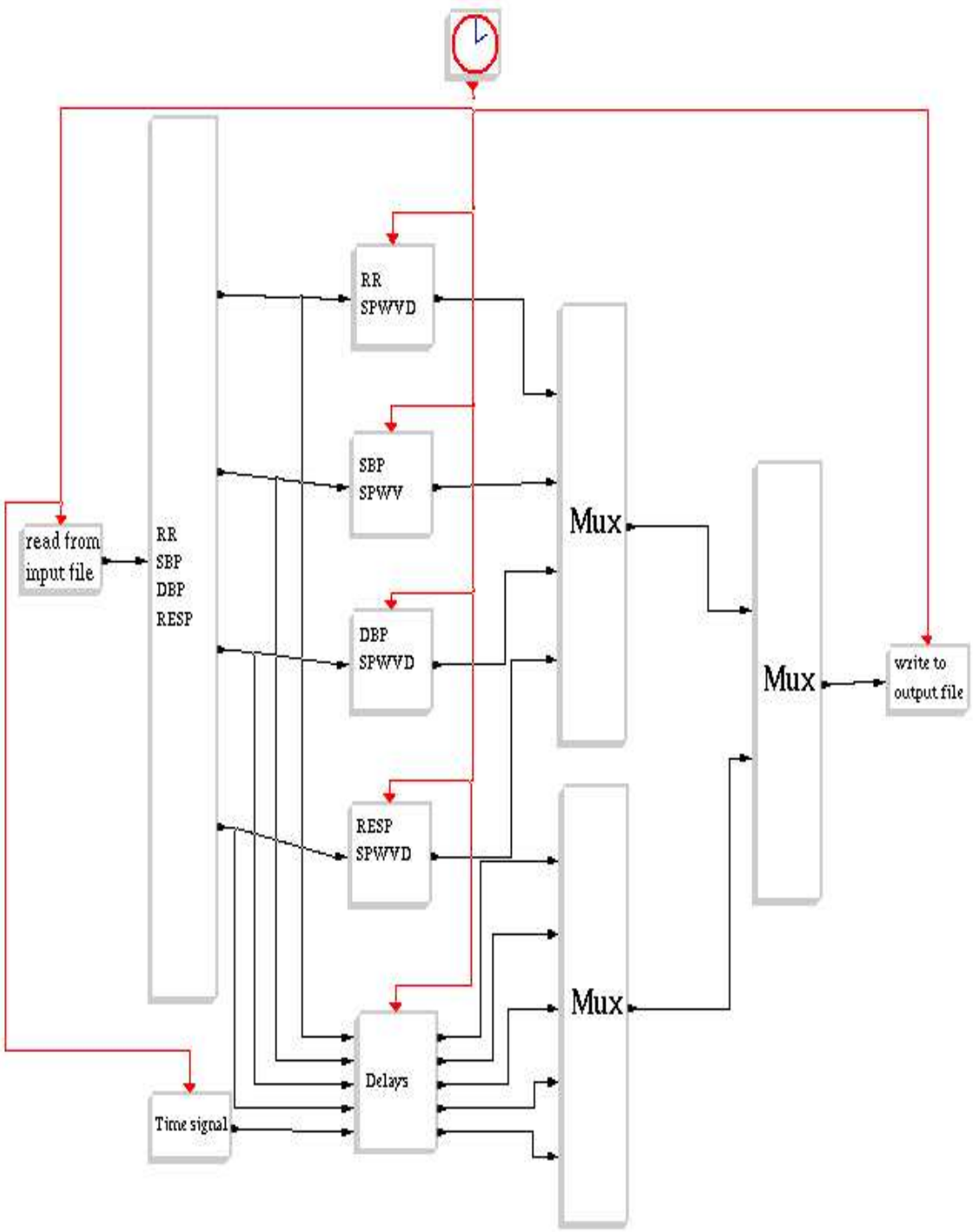


Figure 54: *General view of the SPWVD applied to the cardiovascular and respiratory signals*

```
freq_sampling=4;
cstRR=1000;cstSBP=110;cstDBP=60;
*****************************************************************************************************************
 frequency bounds for the different WFIR (in Hz)
maxLF_f=0.12;minLF_f=0.04;maxHF_f=0.18;minHF_f=0.12;maxF=0.4;minF=0.04;                    Pre-processing

 large WFIR
CVlarge_coef=355;CVlarge_delay=(CVlarge_coef-1)/2;
CVlarge_ftype='bp';CVlarge_wtype='hm';
CVlarge_low=minF;CVlarge_high=maxF;
 narrow WFIR for the LF band
CVnarrow_coef=355;CVnarrow_delay=(CVnarrow_coef-1)/2;
CVnarrow_ftype='bp';CVnarrow_wtype='hm';
CVnarrow_low=minLF_f;CVnarrow_high=maxLF_f;
 narrow WFIR for the HF band
CVnarrow2_coef=355;CVnarrow2_delay=(CVnarrow_coef-1)/2;
CVnarrow2_ftype='bp';CVnarrow2_wtype='hm';
CVnarrow2_low=minHF_f;CVnarrow2_high=maxHF_f;
 Hilbert filter
hilbert_coef=87;hilbert_delay=(hilbert_coef-1)/2;hilbert_wtype='hm';
*****************************************************************************************************************
 temporal and frequential  filters
temporal_coef=24;temporal_delay=(temporal_coef-1)/2;temporal_wtype='hm';                   SPWVD
frequency_coef=512;frequential_delay=(frequency_coef-1)/2;frequency_wtype='hm';
frequency_points=1024;indextofreq=0.5/frequency_points*freq_sampling;
zerovect=zeros((frequency_points-frequency_coef)/2,1);
*****************************************************************************************************************
low-pass filter for the instantaneous parameters computation                               Instantaneous parameters
freqfilter_coef=55;freqfilter_delay=(freqfilter_coef-1)/2;
freqfilter_ftype='lp';freqfilter_wtype='hm';
freqfilter_low=minF; freqfilter_high=maxF;
powertoamplitude1=8.6206;powertoamplitude2=8.6206;  gain of the filters
*****************************************************************************************************************
Successive delays related to the different filters
global_delay=freqfilter_delay+CVlarge_delay+CVnarrow_delay+hilbert_delay+1+temporal_delay+frequential_delay;
global_delay_sec=global_delay/freq_sampling;
```

Figure 55: *The whole context of the SPWVD applied to the cardiovascular and respiratory signals*

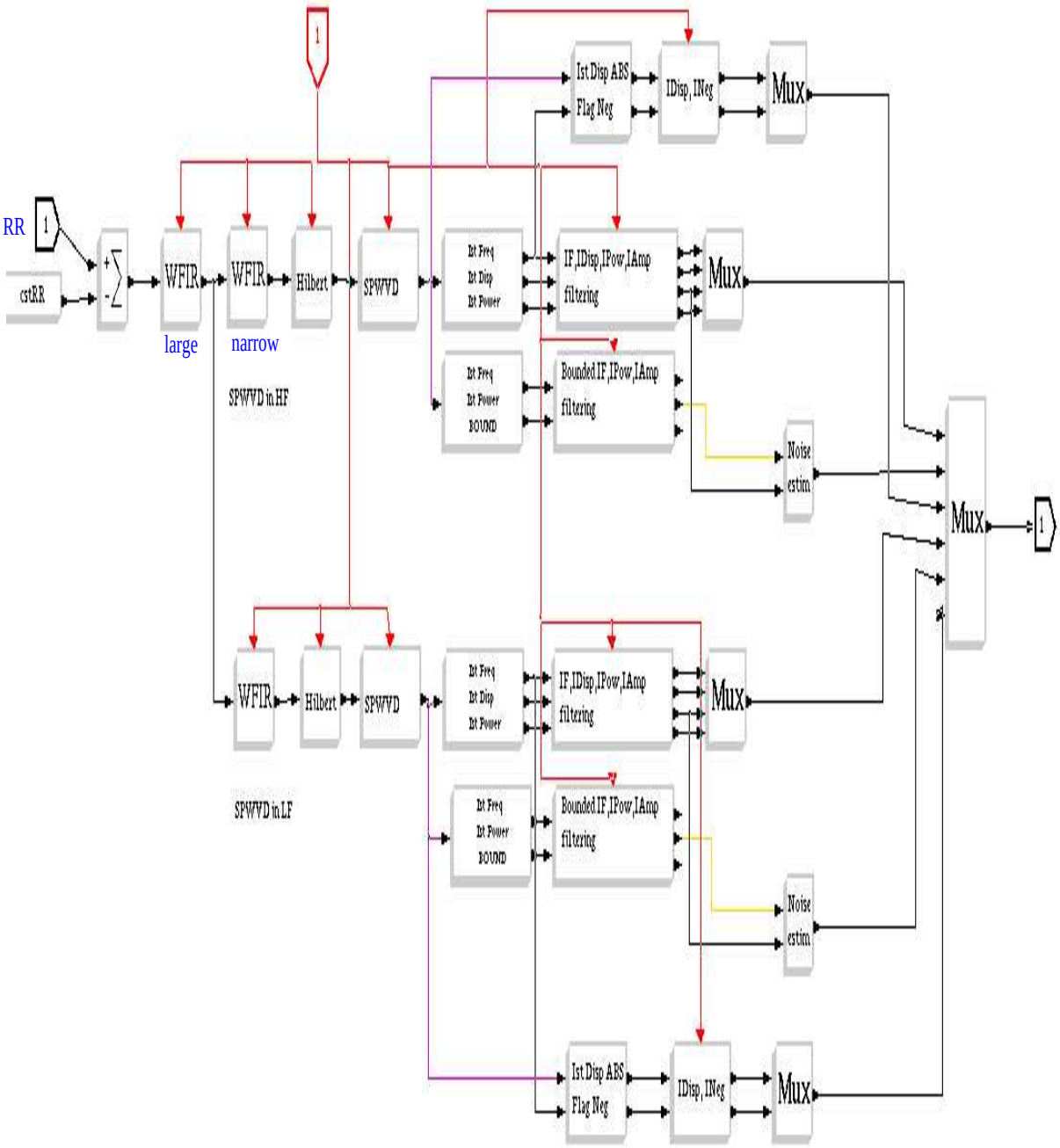


Figure 56: *The flow chart of the SPWVD applied to the RR time series*

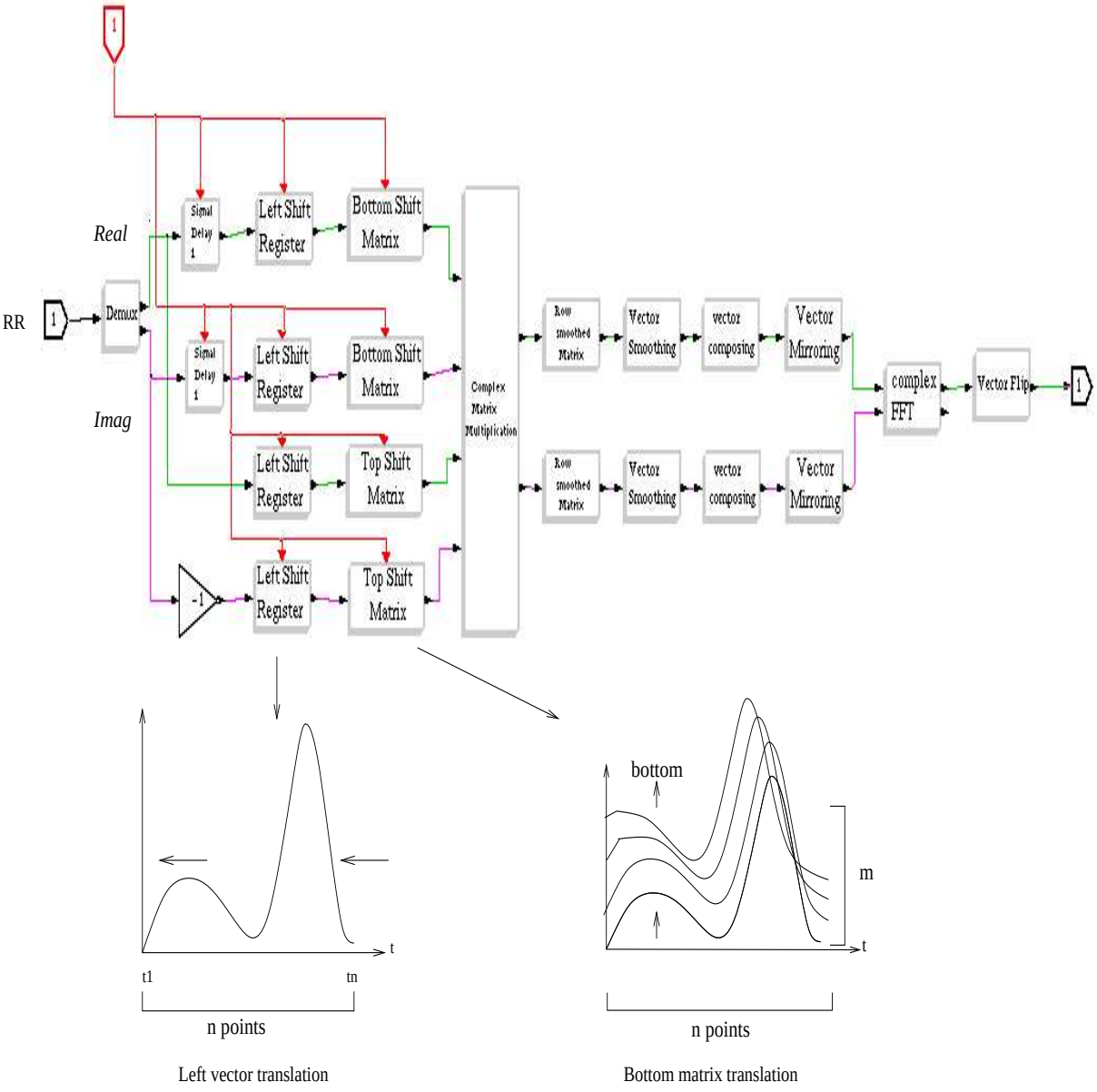


Figure 57: *The SPWVD block*

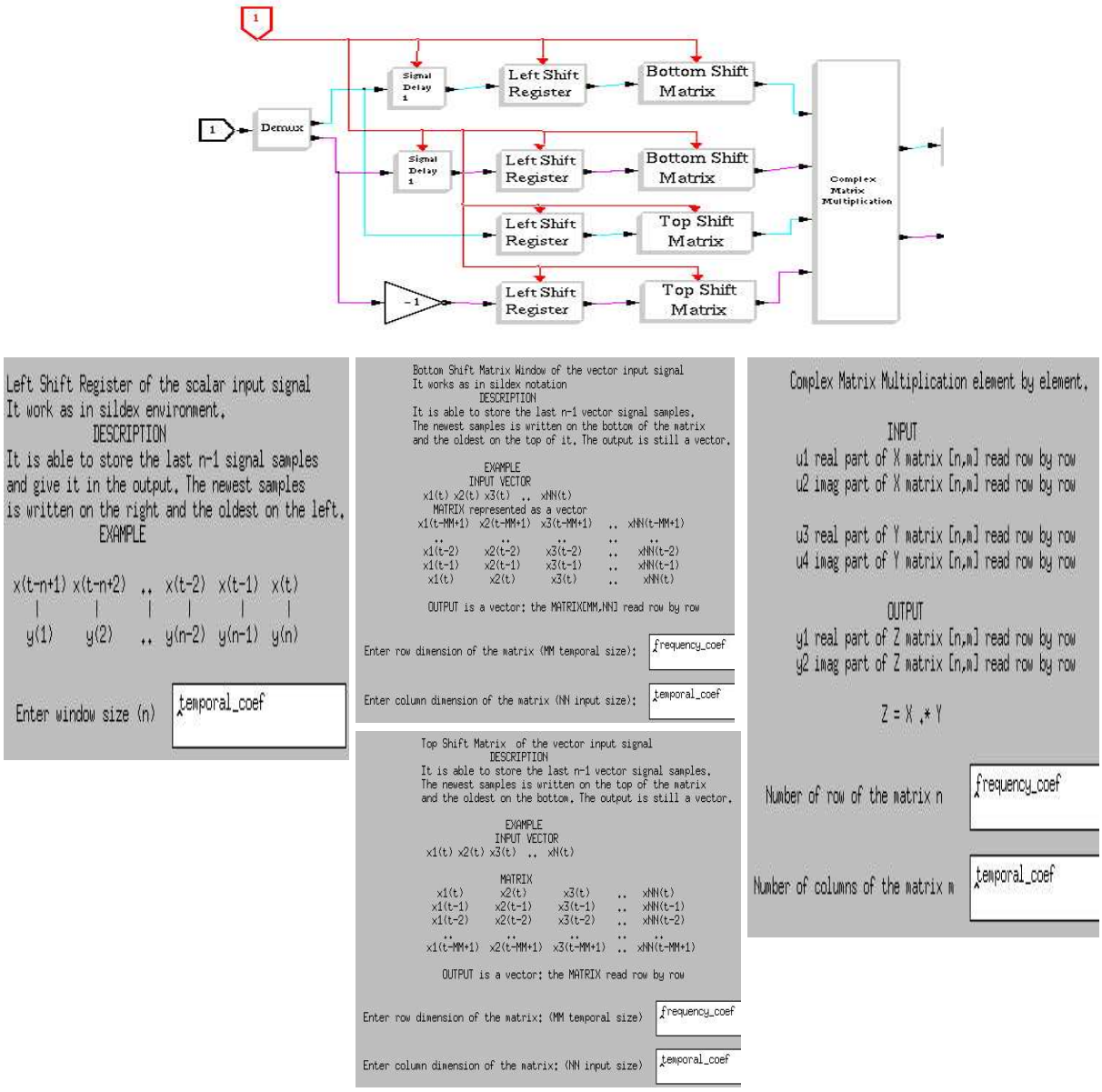


Figure 58: *The parameters of the SPWVD block*

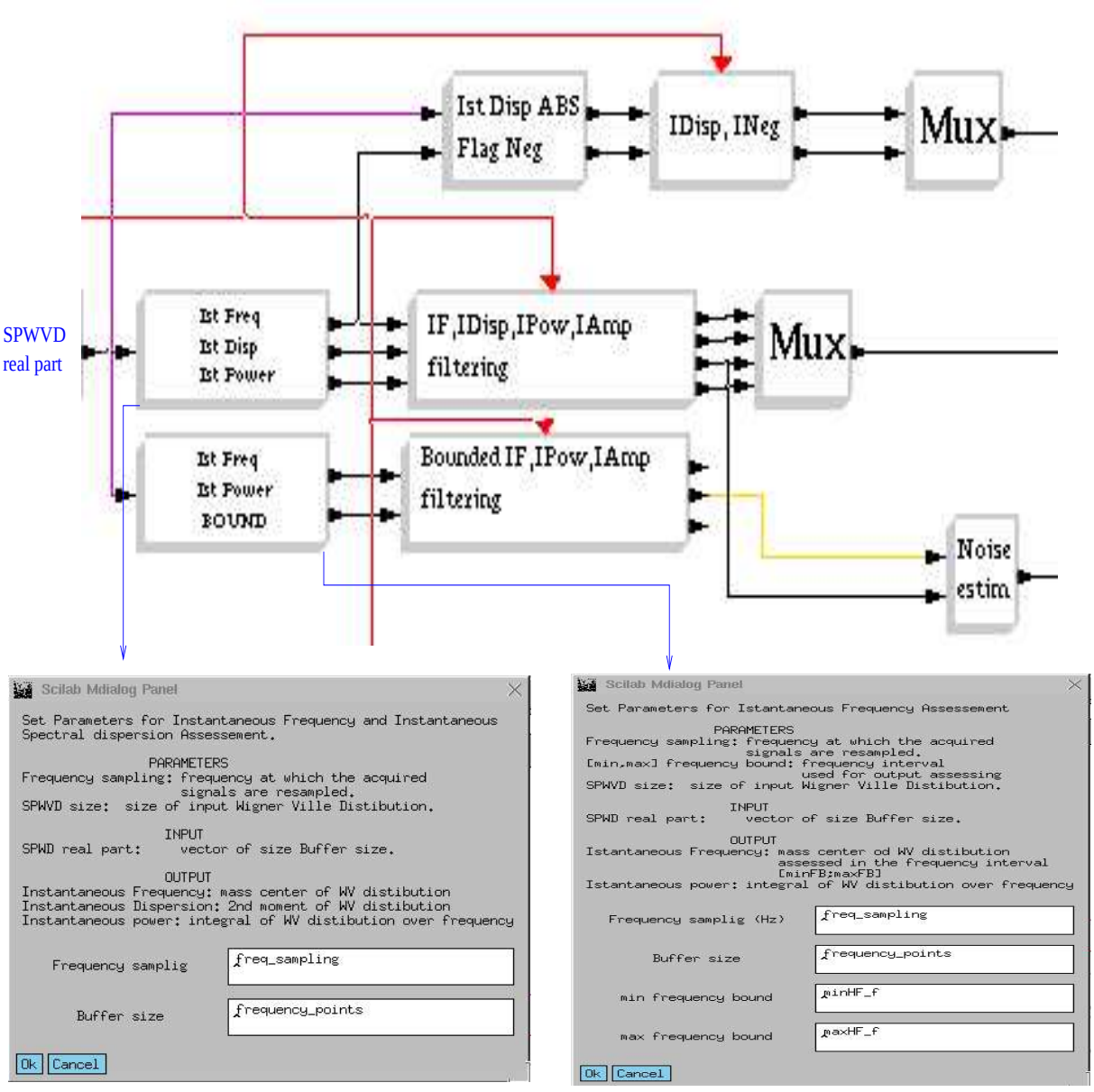


Figure 59: *Computing of the instantaneous frequency and power: The Ist Freq, Ist Power block*

5.4.2 The 3D visualisation of the SPWVD

SCICOS implementation of the SPWVD for a 3D visualisation: we can use a simplified version of the SCICOS program, which does not compute the instantaneous parameters, frequency, power and amplitude, but which simply writes the raw matrix of the successive FFT on an output file. The figures 60 and 61 are respectively dedicated to the SPWVD in HF and LF band, and to the SPWVD of the whole spectrum between 0.04 - 0.4 Hz.

SCILAB implementation of the SPWVD visualisation: The *SPWVfor3Dvisu.sci* file, represented on the figure 62, simply uses the predefined *plot3d1* function, with some options to adapt the parameters of visualisation. A point has to be taken into account: when using the output of a SCICOS program, remember that the first column is dedicated to the SCICOS simulation clock. In addition, the second column here is dedicated to the delayed time, allowing to keep an exact synchronisation with others signals.
The 3D visualisation of the SPWVD of RR, SBP and respiratory time series during an head-up 60 degrees is represented on the figure 63.

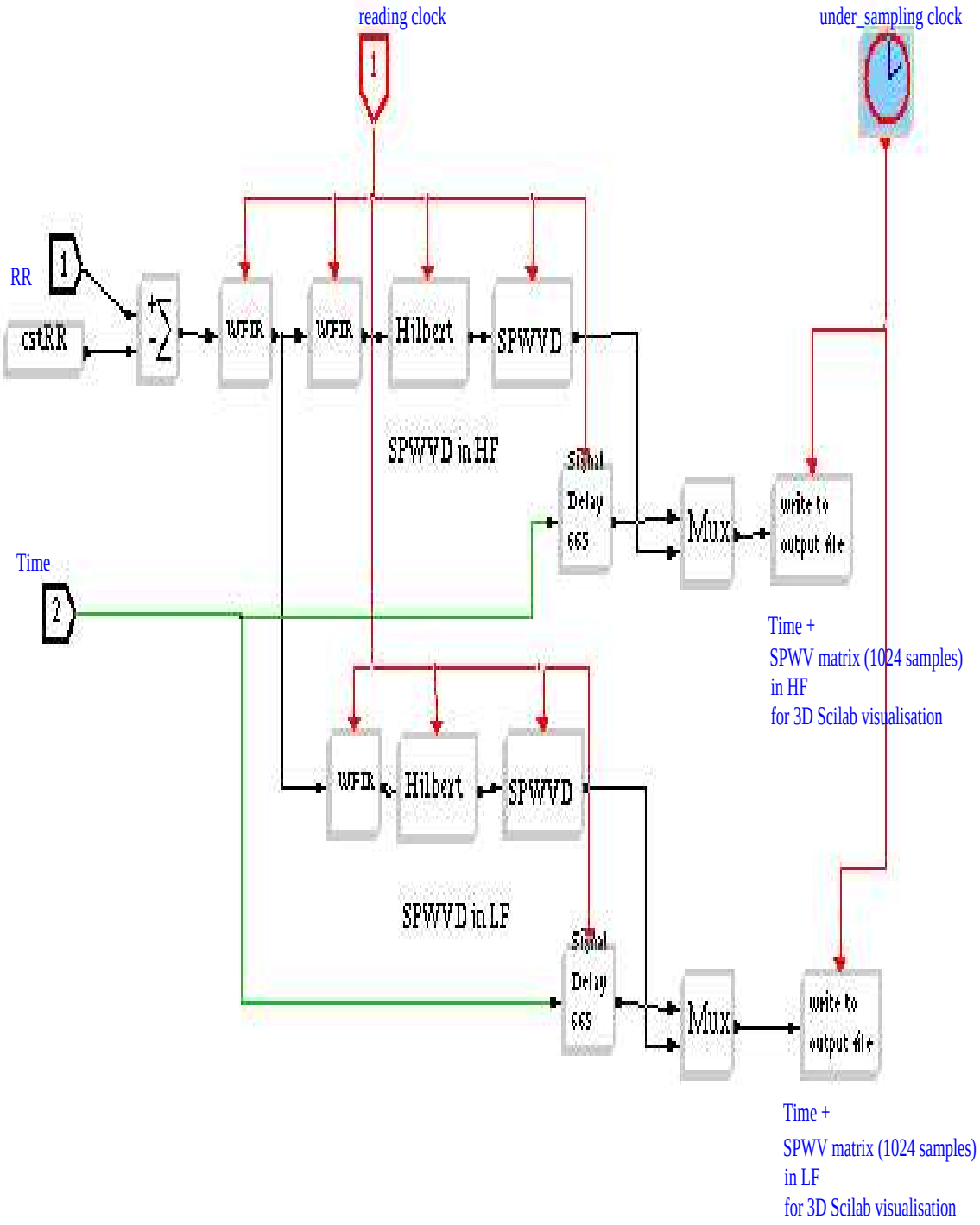


Figure 60: *The SPWVD application in SCICOS dedicated to the 3D visualisation in the HF and in the LF band*

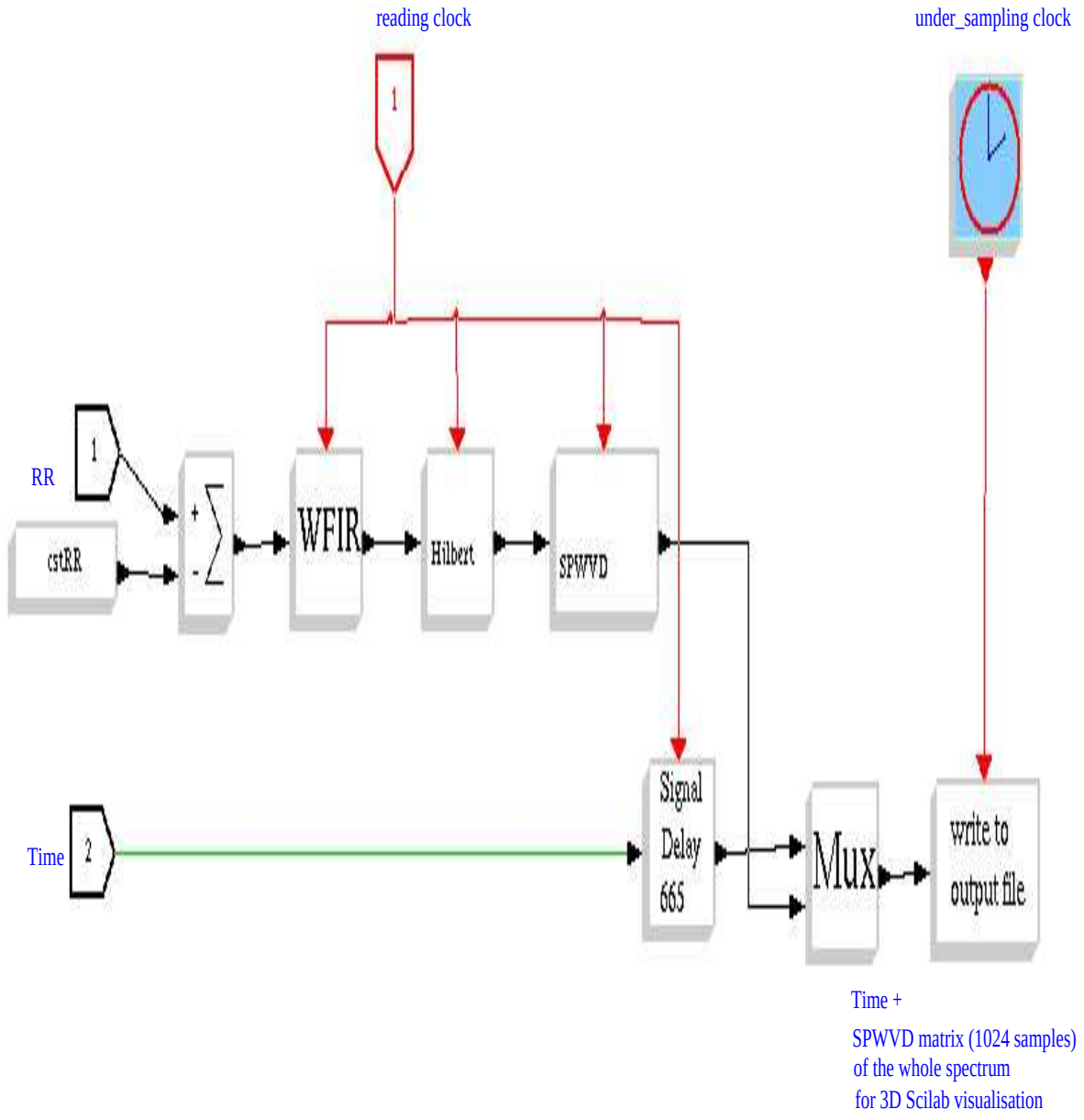
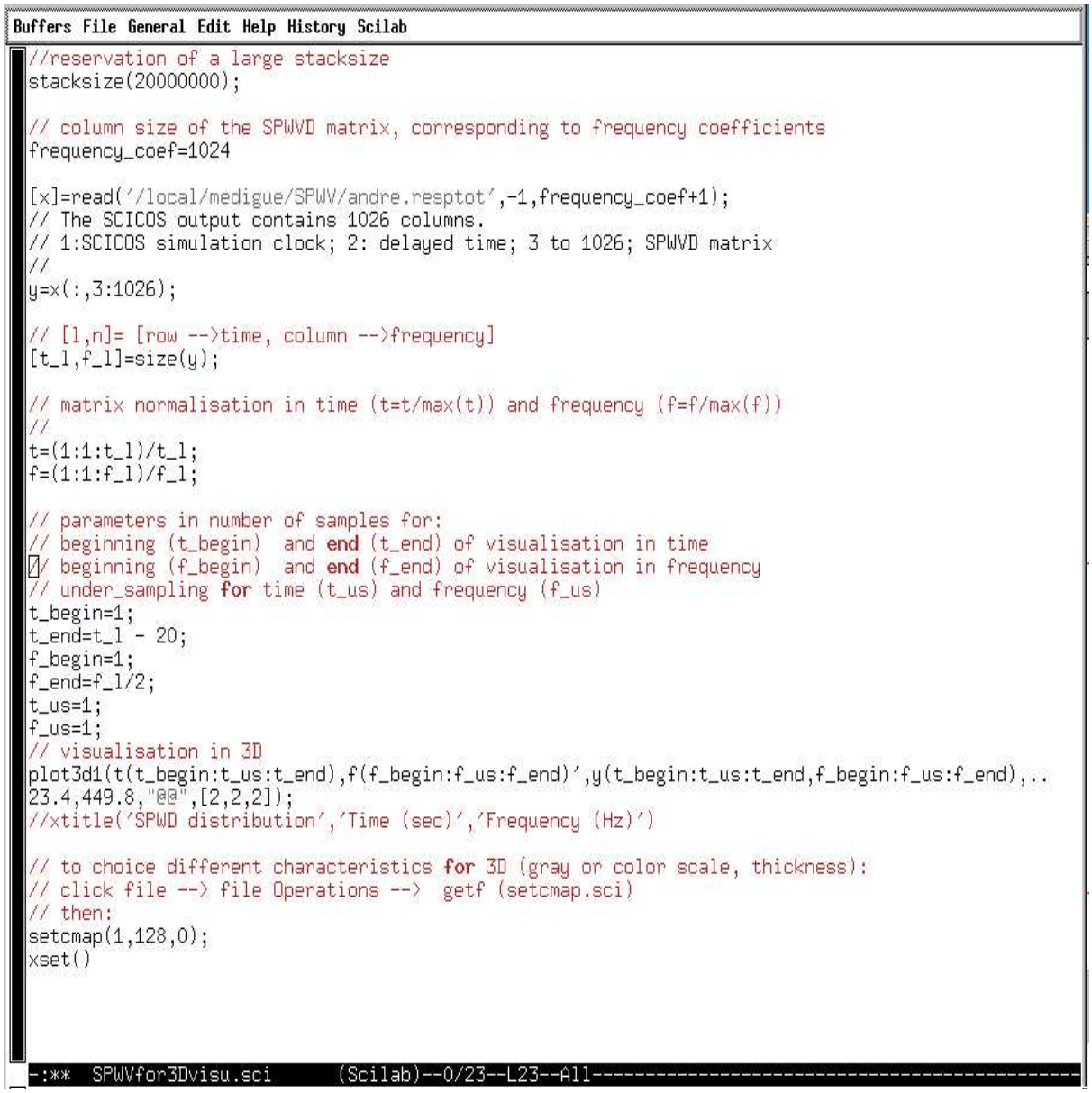


Figure 61: *The SPWVD application in SCICOS dedicated to the 3D visualisation of the whole spectrum*



```
Buffers File General Edit Help History Scilab
//reservation of a large stacksize
stacksize(20000000);

// column size of the SPWVD matrix, corresponding to frequency coefficients
frequency_coef=1024

[x]=read('/local/medigue/SPWV/andre.resptot',-1,frequency_coef+1);
// The SCICOS output contains 1026 columns.
// 1:SCICOS simulation clock; 2: delayed time; 3 to 1026; SPWVD matrix
//
y=x(:,3:1026);

// [l,n]= [row -->time, column -->frequency]
[t_l,f_l]=size(y);

// matrix normalisation in time (t=t/max(t)) and frequency (f=f/max(f))
//
t=(1:1:t_l)/t_l;
f=(1:1:f_l)/f_l;

// parameters in number of samples for:
// beginning (t_begin)  and end (t_end) of visualisation in time
// beginning (f_begin)  and end (f_end) of visualisation in frequency
// under_sampling for time (t_us) and frequency (f_us)
t_begin=1;
t_end=t_l - 20;
f_begin=1;
f_end=f_l/2;
t_us=1;
f_us=1;
// visualisation in 3D
plot3d1(t(t_begin:t_us:t_end),f(f_begin:f_us:f_end)',y(t_begin:t_us:t_end,f_begin:f_us:f_end),..
23.4,449.8,"@@",[2,2,2]);
//xtitle('SPWD distribution','Time (sec)','Frequency (Hz)')

// to choice different characteristics for 3D (gray or color scale, thickness):
// click file --> file Operations -->  getf (setcmap.sci)
// then:
setcmap(1,128,0);
xset()

-:**  SPWVfor3Dvisu.sci      (Scilab)--0/23--L23--All----------------------------------------------
```

Figure 62: *The 3D visualisation SCILAB function*

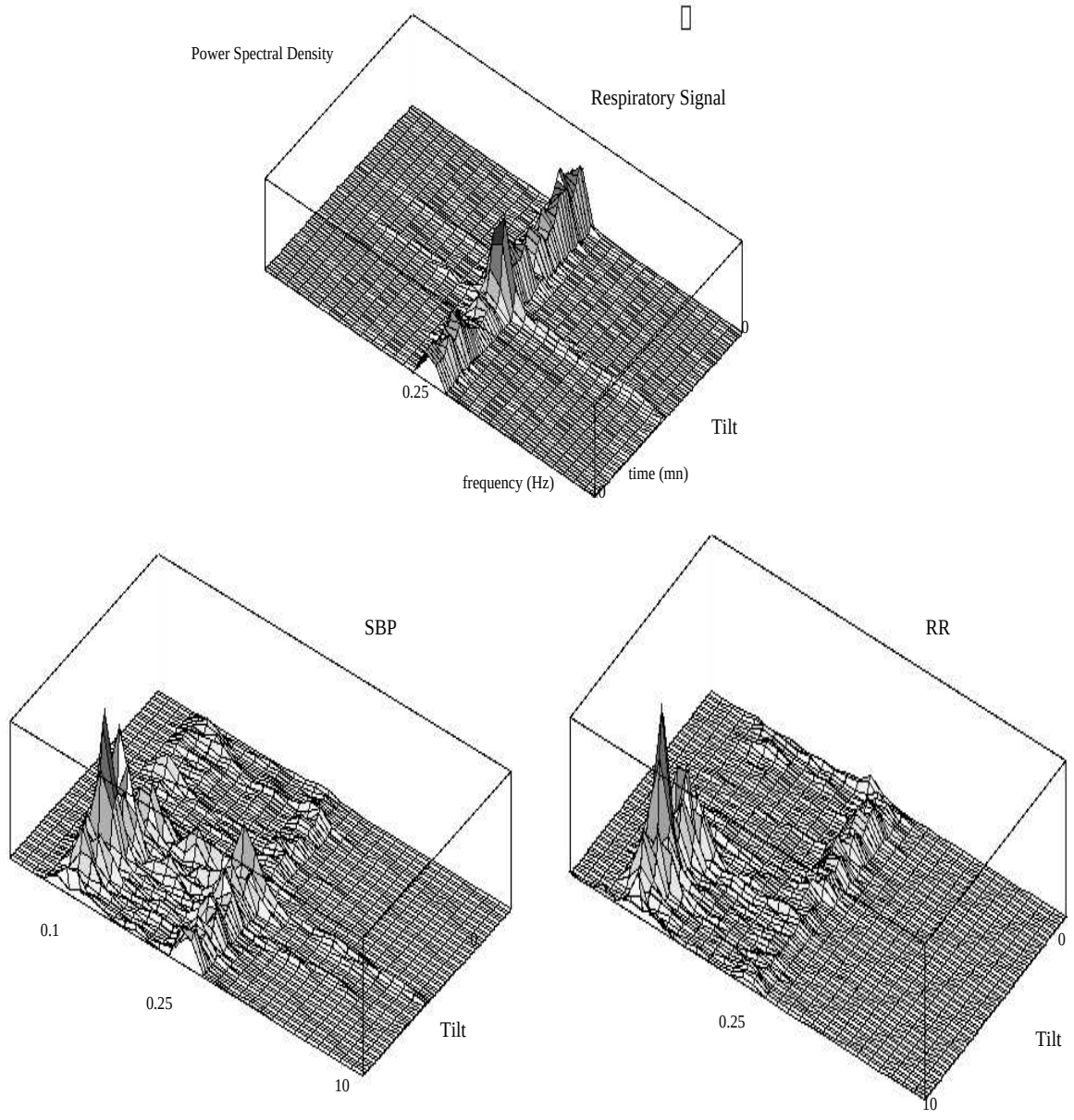


Figure 63: *Graphical result of the 3D visualisation programs*

6 Time analysis of cardiovascular time series

6.1 Presentation

This section is limited to a time method for studying the Baroreflex Sensitivity, and represents an alternative to the spectral Gain method previously described. Spontaneous sequences of 3 or more consecutive beats during which RR and SBP time series have simultaneous increases or decreases were called Baroreflex Sequences.

6.2 The Sequences Method

Such features research uses the SCICOS program called *sequencesRRSBP.cos*, shown on the figure 64. The inputs, RR and SBP times series have to be resampled at the mean frequency of the RR series, to keep a perfect synchronization. Indeed, due to the calibration of the Finapres, the arterial blood pressure signal is lost from time to time and has to be reconstructed by a spline function, as seen previously. The *sequences* block searchs a common increasing or decreasing sequence of successive RR and SBP samples, in a given window of N_seq lenght. The result is a vector of $(N_seq + 1) * 2$ lenght, corresponding to the following variables, as shown on the figure 65: the time, expressed in seconds, corresponding to the SCICOS simulation clock; the type of the sequence, increasing (1) or decreasing (-1); the number of samples in a sequence; the values of theses samples, RR then of SBP (Fig. 65).
This program detects all the sequences, without taking into account the criteria of validity, as for example a rise or fall of 5 or more mmHg for the SBP and of 3 or more bpm for the cardiac rhythm. The user has to choice himself the periods of the cardiovascular time series to detect sequences, then, he has to select the sequences corresponding to the usual criteria.

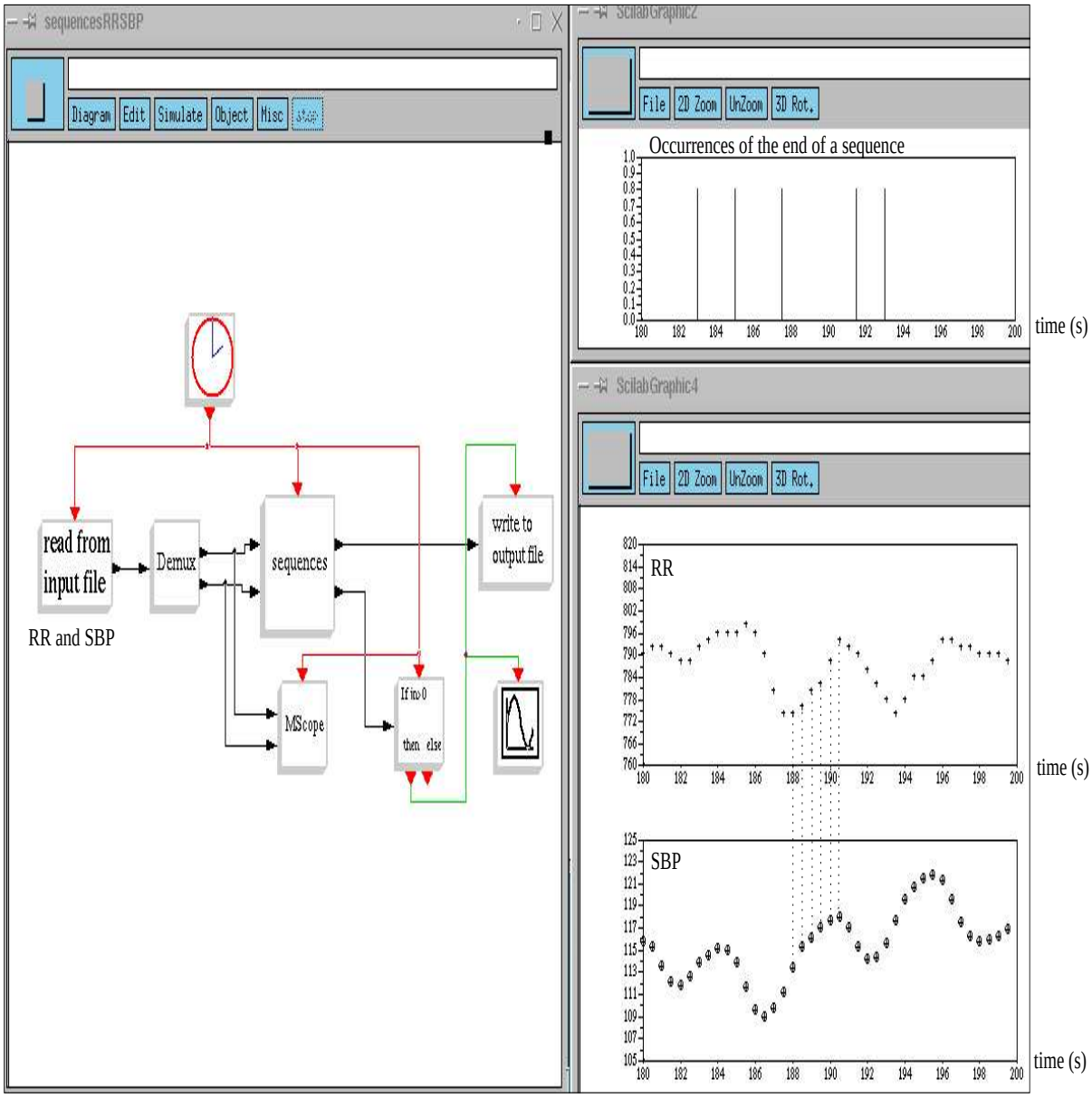


Figure 64: *The sequences detection from RR and SBP time series*

Extract of the output file for the sequence lasting at the second 191.5

Time (s)	Direction 1 ↗ -1 ↘	number of samples in the sequence	N_seq reserved places (10)	
191.5	1	6	774.0 776.0 780.0 782.0 788.0 794.0 0 0 0 0	RR (ms)
			113.3 115.3 116.1 117.0 117.7 117.9 0 0 0 0	SBP (mmHg)

scicos clock y1 (1) y1(2) y1(3 ..N+3) for RR and y1(N+4 .. 2*N+2) for SBP

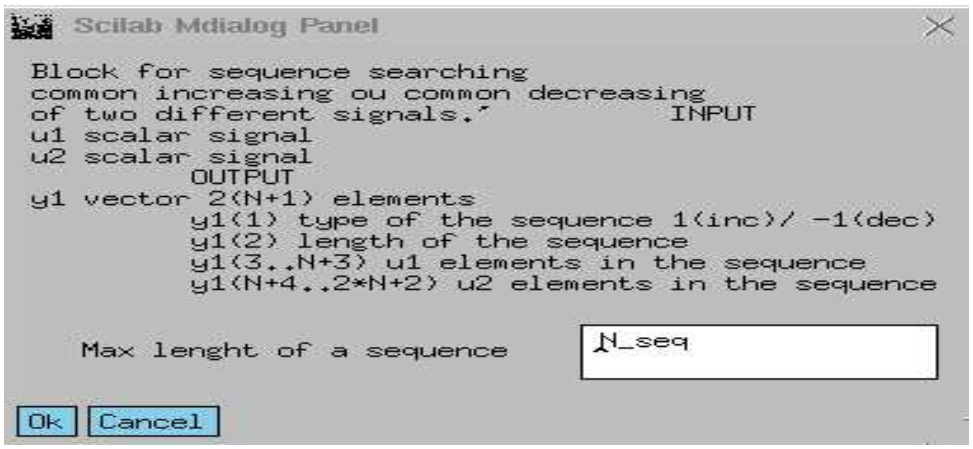


Figure 65: *Example of a sequence detected before the second 191.5*

7 Practical aspects of LARY_CR

7.1 The directories hierarchy

The LARY_CR hierarchy is shown on the figure 66. The *.scilab* file and the LARY_CR directory stand at the login level.

The *.scilab* file is needed for taking into account the functions used in the SCICOS applications. It gets the SCILAB functions and links them with the C and F functions, at the start of SCILAB.

The LARY_CR directory refers to the different kinds of programs used by the applications.

- The *BLOCKS* directory contains the C functions (in the *C_BLOCKS* directory), the F functions (in the *F_BLOCKS* directory), the SCILAB functions (in the *SCI_BLOCKS* directory). Each of the *C_BLOCKS* and *F_BLOCKS* directories include a Makefile. The *SCI_BLOCKS* includes the SCICOS-SCILAB interface functions and the SCILAB functions. The SCICOS-SCILAB interface functions are named as nom_f.sci, the SCILAB functions, as nom.sci. The *PALETTE* directory contains the SCICOS program CVR_blocks.cos, composed of the elementary blocks dedicated to the cardiovascular signal processing.

- The *DEMODULATION*, *DETECTION*, *FFT*, *GAIN*, *SEQUENCES*, *SPWVD* directories are dedicated to the SCICOS programs.
  The *DETECTION* directory includes the detection programs of the RR and BP time series, and a program named detectionRawResp.cos which only delays the respiratory signal by the delays of the filters used in the detection of RR and BP.
  In the *DEMODULATION* directory, 3 examples are done, built on the same scheme. The alone change between them concerns the frequency band used for demodulating. In particular, CDM_025.cos is applied to a paced breathing condition at 0.25 Hz and CDM_spont.cos is applied to a spontaneous breathing condition. These examples could help the user to adapt the CDM program to its own application.
  The *GAIN* directory contains also 3 examples built on the same scheme. The gainRRBP.cos program computes the coherence and gain between RR and SBP, RR and DBP, whereas the two others compute also the coherence between the respiratory signal and the cardiovascular signals. gainRRBP_pacedRESP.cos is applied to a regular breathing condition, whereas gainRRBP_spontRESP.cos is applied to a spontaneous, irregular breathing condition. As previously seen, in the first case, the HF band is centered around the central respiratory frequency. In the second case, the HF band is a priori defined.
  The *FFT* directory contains the FFT programming of the RR and SBP series and the visualization of their spectrum. One can modifie this program by adding the computation of the spectral values in each frequency band.
  The *SEQUENCES* directory contains one program: sequencesRRSBP.cos.
  The *SPWVD* directory contains several programs. SPWVD.cos is the whole application, computing the instantaneous parameters. SPWVDfor3D.cos and SPWVDfor3D_global.cos computed only the instantaneous spectrum, SPWVDfor3D.cos computes the spectrum in the HF and in the LF band separately whereas SPWVDfor3D_global.cos computes the

whole spectrum between 0.04 and 0.4 z. The visualization of the resulting matrix is done by the SPWVDfor3Dvisu.sci.

- The *man* directory contains the man files used by the SCILAB help. An example is done on the figure 67. When creating a new block, it is possible to comment it by adding a man file in the *man* directory. Then one has to compile it by using the SCILAB *formatman* function in the SCILAB window.

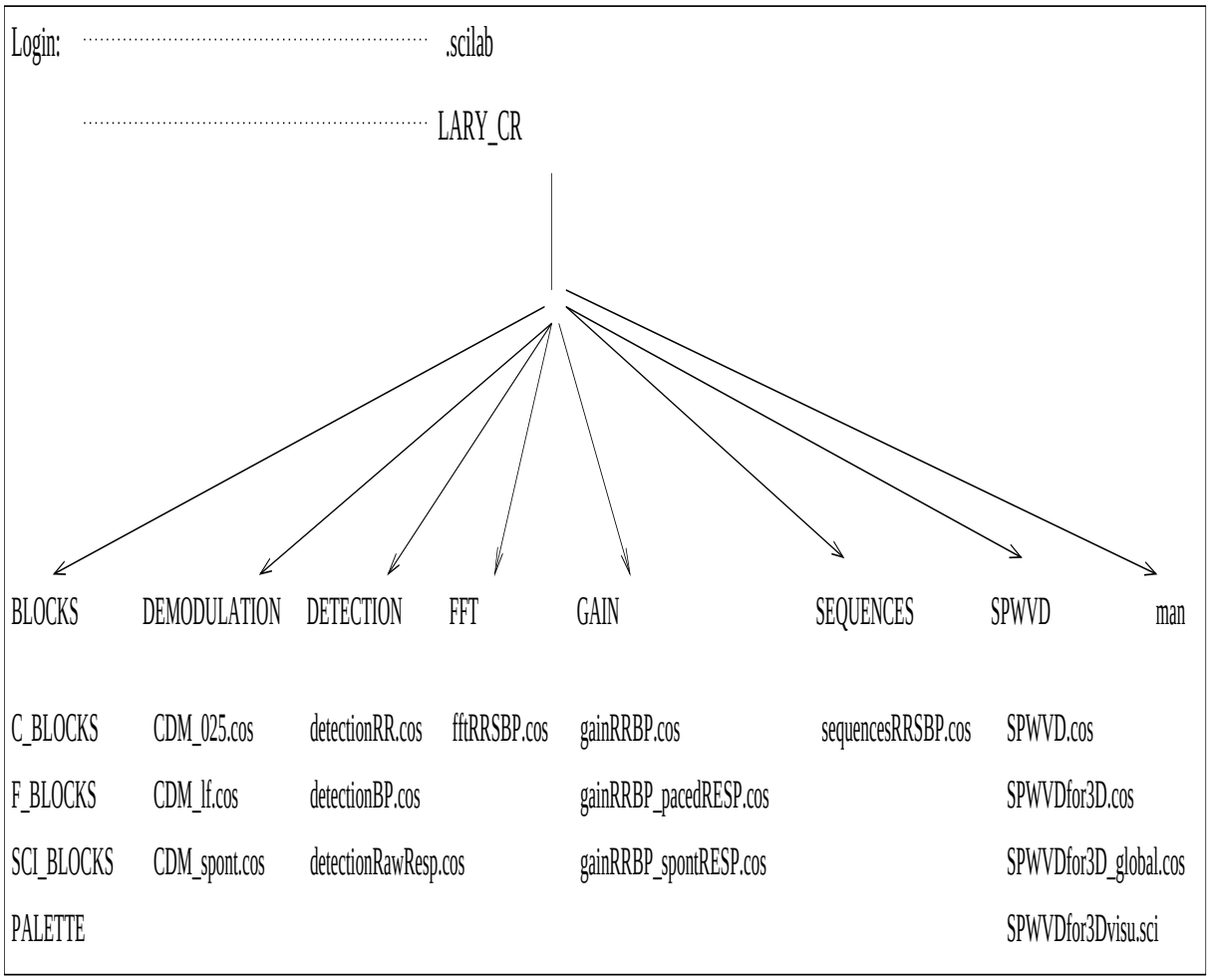


Figure 66: *The LARY_ CR hierarchy*

7.2 Install

First, down load the SCILAB package at:
http://www-rocq.inria.fr/scilab.

Secondly, down load the LARY_CR package at:
ftp://ftp.inria.fr/INRIA/Projects/sosso/LARY_CR/LARY_CR.tar.gz

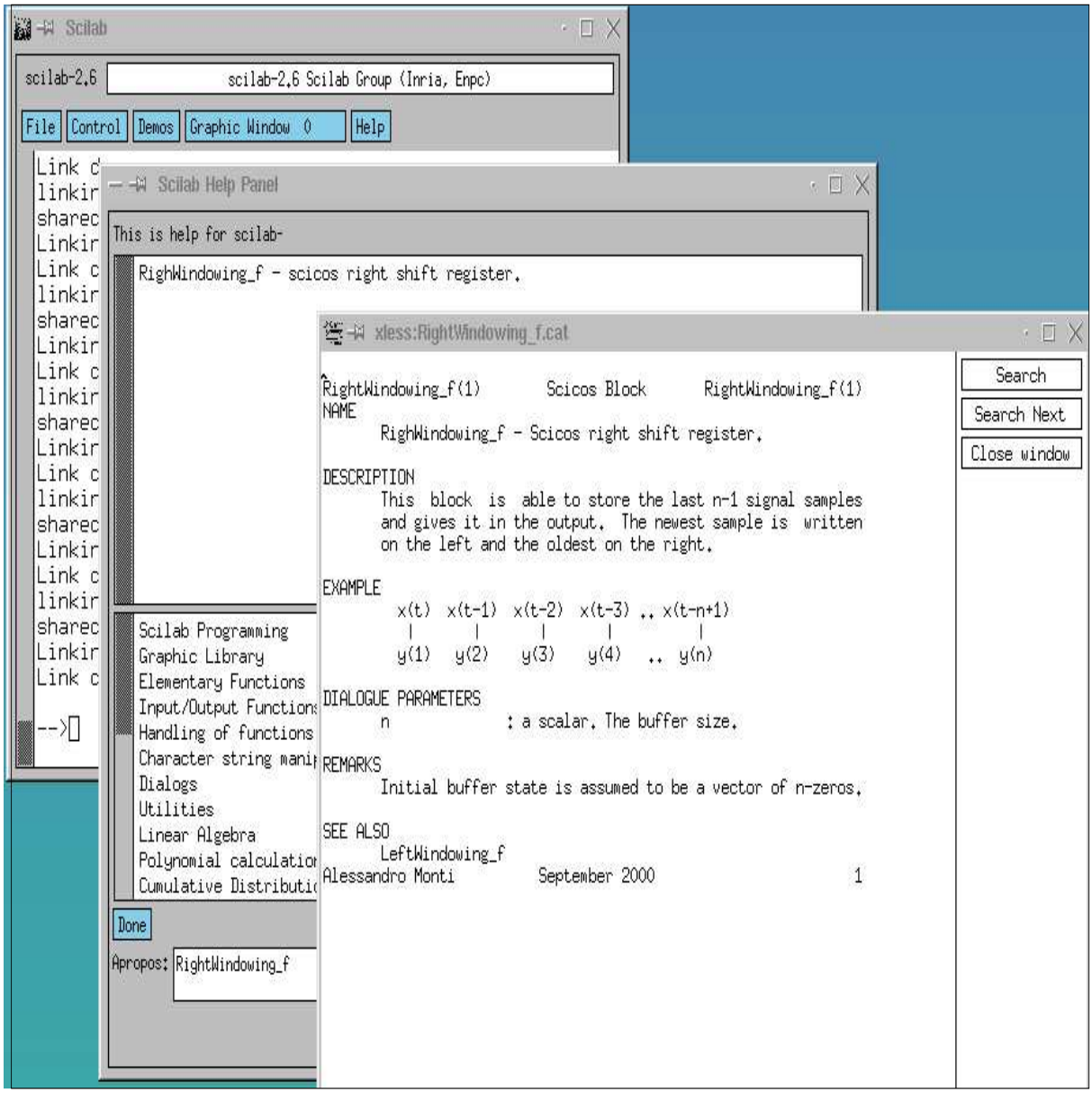


Figure 67: *The man file for a LARY_CR block, the RightWindowing*

Bring up to date the user PATHS in the .scilab file and in the **Pal Editor** window of the SCICOS main diagram. The **Pal Editor** window contains the PATHS of the SCICOS palettes, in particularly the PATH of the CVR_blocks palette. One can access to the **Pal Editor** by clicking in the **Edit** menu of the main SCICOS window.
Compile the C and F functions by executing their Makefile.
When starting a LARY_CR application in the SCICOS environment, the diagram can overflow the size of the window ; one has to adjust it by changing the **zoom** option in the **Misc** menu of the main SCICOS window.

7.3 How to create or modify a program in the SCICOS environment?

7.3.1 At the SCICOS graphical level

When using the pre-existing elementary blocks, an application can be completely build, or simply modified by adding and linking blocks extracted from the **Palettes**.

7.3.2 At the F and C functions level

To create a new elementary block, one has to act at different levels. First, create the new F or C function named *nom.f* in the *F_BLOCKS* directory for example, add it to the Makefile and compile. Secondly, create the interface function named *nom_f.sci* in the *SCI_BLOCKS* directory. Thirdly, add them into the .scilab file and restart SCILAB. To test the new block, start SCICOS and call the **Add new block** option in the **Edit** menu of the main SCICOS window. Give the block name as *nom_f*, the block is directly copied in the untitled window. When all is right, one can add this new block to the CVR_blocks palette by loading the CVR_blocks program and adding the new block in it.

Unité de recherche INRIA Rocquencourt
Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38330 Montbonnot-St-Martin (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-0803